

**BALANCING CONVERGENCE AND DIVERSITY IN EVOLUTIONARY SINGLE,
MULTI AND MANY OBJECTIVES**

By

Haitham Seada

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science – Doctor of Philosophy

2017

ABSTRACT

BALANCING CONVERGENCE AND DIVERSITY IN EVOLUTIONARY SINGLE, MULTI AND MANY OBJECTIVES

By

Haitham Seada

Single objective optimization targets only one solution, that is usually the global optimum. On the other hand, the goal of multiobjective optimization is to represent the whole set of trade-off Pareto-optimal solutions to a problem. For over thirty years, researchers have been developing Evolutionary Multiobjective Optimization (EMO) algorithms for solving multiobjective optimization problems. Unfortunately, each of these algorithms were found to work well on a specific range of objective dimensionality, i.e. number of objectives. Most researchers overlooked the idea of creating a cross-dimensional algorithm that can adapt its operation from one level of objective dimensionality to the other. One important aspect of creating such algorithm is achieving a careful balance between convergence and diversity. Researchers proposed several techniques aiming at dividing computational resources uniformly between these two goals. However, in many situations, only either of them is difficult to attain. Also for a new problem, it is difficult to tell beforehand if it will be challenging in terms of convergence, diversity or both. In this study, we propose several extensions to a state-of-the-art evolutionary many-objective optimization algorithm – NSGA-III. Our extensions collectively aim at (i) creating a unified optimization algorithm that dynamically adapts itself to single, multi- and many objectives, and (ii) enabling this algorithm to automatically focus on either convergence, diversity or both, according to the problem being considered. Our approach augments the already existing algorithm with a niching-based selection operator. It also utilizes the recently proposed Karush Kuhn Tucker Proximity Measure to identify ill-converged

solutions, and finally, uses several combinations of point-to-point single objective local search procedures to remedy these solutions and enhance both convergence and diversity. Our extensions are shown to produce better results than state-of-the-art algorithms over a set of single, multi- and many-objective problems.

To my Beloved Family

ACKNOWLEDGEMENTS

I would like to thank my family especially my wife, Hayam, and my parents for many things that go beyond listing.

I would also like to express my sincere gratitude to my advisor Prof. Kalyanmoy Deb for his support, patience, motivation and immense knowledge. I learnt from him a lot and could not have imagined a better advisor.

A big "Thank You!" also goes to all my COIN lab colleagues. It has been an honor working with each of them. A PhD is a lot easier when you are surrounded with friends.

Last but not least, I would like to thank Fulbright commision in Egypt, AMIDEast Washington D.C. and Beacon Center for making my graduate studies financially feasible. The three agencies shared the same friendly yet professional attitude that made my life a lot easier.

TABLE OF CONTENTS

LIST OF TABLES	ix
LIST OF FIGURES.....	xi
LIST OF ALGORITHMS	xvi
Chapter 1 Introduction.....	1
1.1 Problem Definition	2
1.2 Motivation.....	3
1.3 Thesis Statement	6
1.4 Contribution	6
1.4.1 Unified EMO for Single, Multiple and Many Objectives.....	7
1.4.2 Selection in NSGA-III.....	8
1.4.3 Enhancing Convergence in EMO.....	8
1.4.4 Balancing Convergence and Diversity	9
1.5 Summary	10
Chapter 2 Related Work.....	11
2.1 Introduction.....	11
2.2 Evolutionary Multiobjective Optimization (EMO)	12
2.3 Introduction to NSGA-III	18
2.4 Convergence versus Diversity in EMO.....	21
2.4.1 Towards a Better Balance	21
2.4.2 Optimality in EMO.....	25
2.4.3 Local Search	27
2.5 Road Map	31
2.6 Summary	31
Chapter 3 A Unified Evolutionary Optimization Procedure for Single, Multiple, and Many Objectives	33
3.1 NSGA-III for Single and Multi-objective Problems.....	34
3.2 Proposed Unified Approach: U-NSGA-III.....	37
3.2.1 U-NSGA-III for Single-objective Problems	41
3.2.2 U-NSGA-III for Multi-objective Problems.....	42
3.2.3 U-NSGA-III for Many-objective Problems	43
3.3 Results	44
3.3.1 Single-objective Problems.....	44
3.3.1.1 Unconstrained Problems	45
3.3.1.2 Constrained Problems	48
3.3.2 Bi-objective Problems.....	49
3.3.2.1 N Equal to H	50
3.3.2.2 N Greater Than H	52
3.3.3 Three-objective Problems	54

3.3.4	Many-objective Problems	57
3.4	Summary	59
Chapter 4 Effect of Selection Operator on NSGA-III in Single, Multi, and Many-Objective Optimization..... 62		
4.1	Hypotheses	63
4.1.1	NSGA-III Convergence	64
4.1.2	NSGA-III and Local Optima.....	64
4.2	Results	65
4.2.1	Single-Objective Problems.....	66
4.2.2	Multi and Many-Objective Problems.....	67
4.2.3	Multi-modal Problems.....	73
4.3	Summary	75
Chapter 5 Towards Faster Convergence of Evolutionary Multi-Criterion Optimization Algorithms using Karush-Kuhn-Tucker Optimality Based Local Search... 76		
5.1	A Brief Review of KKTPM NSGA-III	77
5.1.1	Exact KKT Proximity Measure.....	77
5.1.2	Approximate KKTPM Computation Method	80
5.1.2.1	Adjusted KKTPM Computation Method	82
5.1.2.2	Projected KKTPM Computation Method	82
5.1.2.3	Estimated KKTPM Computation Method	82
5.2	Proposed KKTPM based Hybrid Local Search-EMO	83
5.2.1	Identification of a Population Member for Local Search	83
5.2.2	Local Search Procedure	85
5.2.3	Overall Proposed Algorithm	86
5.2.4	Frequency of Local Search.....	87
5.3	Results	87
5.3.1	A Variable-Density Problem.....	88
5.3.2	Bi-Objective Test Problems.....	90
5.3.2.1	Specific Test Problems.....	91
5.3.2.2	Three and Many-objective DTLZ Problems	103
5.3.2.3	Engineering Design Problems	105
5.4	Summary	107
Chapter 6 Multi-Phase Balance of Diversity and Convergence in Multiobjective Optimization109		
6.1	Proposed B-NSGA-III	110
6.1.1	Alternating Phases.....	113
6.1.2	Two Local Search Operators	118
6.1.2.1	Extreme-LS.....	119
6.1.2.2	Achievement Scalarization Function LS (ASF-LS)	120
6.2	Results	120
6.2.1	Multi-objective problems.....	121
6.2.2	Many-Objective Optimization	128
6.2.3	Using numerical gradients.....	130

6.3 Summary	133
Chapter 7 Conclusions and Future Work	134
7.1 Future Work.....	135
APPENDIX	138
BIBLIOGRAPHY	156

LIST OF TABLES

TABLE 3.1:	Threshold HV values used in bi-objective ZDT problems for $N \geq H$ simulations.	52
TABLE 3.2:	The theoretical hypervolume HV_T for DTLZ1 and DTLZ2 for 3, 5, 8, 10 and 15 objectives ($\epsilon = 0.01$)	55
TABLE 3.3:	Ideal and reference points used for constrained bi-objective problems. .	56
TABLE 3.4:	Best, medium and worst HV(norm) on multi/many-objective unconstrained DTLZ problems.	57
TABLE 4.1:	Parameter listing.....	66
TABLE 5.1:	Parameters - Columns represent – left-to-right – problem name, population size, number of function evaluations, frequency of LS (per generation), maximum allowed number of function evaluations per one LS, number of generations with LS and number of generations without LS	88
TABLE 6.1:	Parameters used by B-NSGA-III. M is the number of objectives. N is population size. SEs stands for Solution Evaluations. α is the frequency explained in Section 6.1 while β is the maximum limit of function evaluations the single objective optimizer (fmincon()) can use. The final column shows the values of ϵ in Equation 6.1.	122
TABLE 6.2:	Each problem test one or more aspects of the algorithm	122
TABLE 6.3:	p-values of a two sided Wilcoxon rank sum test.	122
TABLE A.1:	Single objective unconstrained significance test ($p - value$).....	139
TABLE A.2:	Single objective constrained significance test ($p - value$)	147
TABLE A.3:	Best, medium and worst function values for mono-objective unconstrained test problems.....	149
TABLE A.4:	Best, medium and worst function values for mono-objective constrained test problems.....	149

TABLE A.5: Best, medium and worst HV for bi-objective unconstrained ZDTtest problems.....	150
TABLE A.6: Best, medium and worst HV for bi-objective constrained test problems.	151
TABLE A.7: Best, medium and worst HV on multi/many-objective constrained C3-DTLZ problems.	152
TABLE A.8: (p-values) of Wilcoxon significance test for unconstrained bi-objective test problems	153
TABLE A.9: (p-values) of Wilcoxon significance test for constrained bi-objective problems.....	154
TABLE A.10: (p-values) of Wilcoxon significance test for unconstrained, constrained and scaled multi/many-objectives problems	155

LIST OF FIGURES

FIGURE 2.1:	Schematic Description of VEGA (copied from [1])	13
FIGURE 2.2:	Crowding Distance in NSGA-II (taken from [2])	15
FIGURE 2.3:	NSGA-II Algorithm (taken from [2])	16
FIGURE 2.4:	MOEA/D PBI (taken from [3])	17
FIGURE 2.5:	Niching in NSGA-III (taken from [4])	18
FIGURE 2.6:	Road Map	32
FIGURE 3.1:	Working principles of NSGA-II and NSGA-III.	35
FIGURE 3.2:	Selection in U-NSGA-III.	40
FIGURE 3.3:	Objective function value with respect to the number of function evaluations.	48
FIGURE 3.4:	Performance of NSGA-II, NSGA-III, and U-NSGA-III with $N = H$ on unconstrained bi-objective test problems.	51
FIGURE 3.5:	Performance of NSGA-II, NSGA-III, and U-NSGA-III with $N \geq H$ on unconstrained bi-objective ZDT test problems.	53
FIGURE 3.6:	Performance of NSGA-II, NSGA-III, and U-NSGA-III on scaled unconstrained three-objective DTLZ problems.	56
FIGURE 3.7:	Performance of NSGA-III and U-NSGA-III on scaled unconstrained many-objective DTLZ problems.	59
FIGURE 4.1:	Single-objective Unconstrained Problems.....	68
FIGURE 4.2:	Single-objective Constrained Problems (G1, G2, G4, G6, G7 and G8). .	69
FIGURE 4.3:	Single-objective Constrained Problems (G9, G10, G18 and G24).....	70
FIGURE 4.4:	Bi-objective Unconstrained Problems (GD)	72

FIGURE 4.5:	Bi-objective Unconstrained Problems (fronts)	73
FIGURE 4.6:	Multi-/Many-objective Problems	73
FIGURE 4.7:	Small population size in multi-modal problems.	74
FIGURE 5.1:	Second constraint governs at iterate $x(k)$ which is not a KKT point.....	81
FIGURE 5.2:	Second constraint governs at iterate $x(k)$ which is not a KKT point.....	81
FIGURE 5.3:	Identification of poorly converged non-dominated solution for LS.....	84
FIGURE 5.4:	ASF-based LS performed in the normalized objective space.....	86
FIGURE 5.5:	Pareto-optimal front and relative density of solutions for the varying density problem.	89
FIGURE 5.6:	Proposed LS replaces non-Pareto-optimal solution with a true Pareto-optimal solution.....	89
FIGURE 5.7:	KKTPM comparison with and without LS on problem OSY. The lines with circles are from the LS based method.	91
FIGURE 5.8:	Effect of LS on problem OSY.	91
FIGURE 5.9:	KKTPM comparison with and without LS on BNH.	93
FIGURE 5.10:	Effect of LS on BNH objective space.	93
FIGURE 5.11:	Effect of LS on BNH variable space.	93
FIGURE 5.12:	NSGA-III with and without LS on ZDT4.....	95
FIGURE 5.13:	Leaps enable faster convergence on ZDT4.....	96
FIGURE 5.14:	Effect of LS on bi-objective unconstrained test problems. The hybrid approach shows a better convergence property.	98
FIGURE 5.15:	LS effect on bi-objective constrained test problems SRN and TNK. The hybrid approach shows a better convergence property.....	99

FIGURE 5.16:	Final fronts obtained for ZDT1 and ZDT6 with and without LS. The hybrid approach with LS converges better.....	99
FIGURE 5.17:	Hypervolumes compared with and without LS for bi-objective unconstrained problems.....	100
FIGURE 5.18:	Hypervolumes compared with and without LS for bi-objective constrained problems. In most cases, hypervolume is better for the hybrid approach.	101
FIGURE 5.19:	Effect of LS on three-objective DTLZ problems demonstrating better convergence property.....	103
FIGURE 5.20:	Final fronts of DTLZ1 and DTLZ2, with (right) and without (left) local search, on three objectives obtained with a limited FE.	104
FIGURE 5.21:	Comparable hypervolume for DTLZ1 and DTLZ2 with and without LS on three objective test problems.	105
FIGURE 5.22:	Effect of LS on 10-objective DTLZ problems. The hybrid approach shows a better convergence property.....	106
FIGURE 5.23:	Effect of LS on two engineering design problems. The hybrid approach shows a better convergence property.....	106
FIGURE 6.1:	<i>Phase-1, Phase-2 and Phase-3 in action</i>	114
FIGURE 6.2:	<i>Alternation of Phases</i>	115
FIGURE 6.3:	<i>A typical alternation of phases of B-NSGA-III while solving ZDT4</i>	123
FIGURE 6.4:	<i>Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on bi-objective ZDT3</i>	123
FIGURE 6.5:	<i>Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on bi-objective ZDT4</i>	123
FIGURE 6.6:	<i>Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on bi-objective ZDT6</i>	124
FIGURE 6.7:	<i>Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on bi-objective TNK</i>	124

FIGURE 6.8:	Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on bi-objective OSY.....	125
FIGURE 6.9:	How median KKTPM progresses over generations in ZDT4, ZDT6 and OSY.....	125
FIGURE 6.10:	Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on DTLZ4 (3 objectives).	126
FIGURE 6.11:	Median final fronts of U-NSGA-III, and MOEA/D on DTLZ4 (3 objectives).....	127
FIGURE 6.12:	Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on DTLZ7 (3 objectives).	127
FIGURE 6.13:	Median final fronts of U-NSGA-III, and MOEA/D on DTLZ7 (3 objectives).....	127
FIGURE 6.14:	Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on DTLZ4 (5 objectives).	128
FIGURE 6.15:	Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on DTLZ4 (10 objectives).....	128
FIGURE 6.16:	PCPs of B-NSGA-III, U-NSGA-III, and MOEA/D on DTLZ4 (10 objectives).....	128
FIGURE 6.17:	Performance of B-NSGA-III (exact & numerical) and U-NSGA-III on OSY problem.....	130
FIGURE 6.18:	Performance of B-NSGA-III (using numerical gradients), U-NSGA-III, and MOEA/D on WFG1 problem.....	131
FIGURE 6.19:	IGD of the three algorithms on WFG1	131
FIGURE 6.20:	GD of the three algorithms on WFG1	131
FIGURE A.1:	Performance comparison between EliteRGA, NSGA-III, U-NSGA-III and CMA-ES on Rastrigin's unconstrained multimodal problem	139
FIGURE A.2:	Performance of EliteRGA, NSGA-III, U-NSGA-III and CMA-ES on Zakharov's unconstrained test problem	140

FIGURE A.3:	Effect of increasing polynomial mutation distribution index on Ackley's function.....	140
FIGURE A.4:	Effect of increasing polynomial mutation distribution index on Rastigrin's function	141
FIGURE A.5:	Single objective constrained optimization problems (G1, G2, G6, G8, G18 and G24)	141
FIGURE A.6:	Additional single objective constrained optimization problems (G4, G9 and G10).....	142
FIGURE A.7:	Performance of NSGA-II, NSGA-III, and U-NSGA-III with $N = H$ on unconstrained bi-objective test problems	142
FIGURE A.8:	Performance of NSGA-II, NSGA-III, and U-NSGA-III with $N = H$ on constrained bi-objective test problems.....	143
FIGURE A.9:	Performance comparison between NSGA-II, NSGA-III and U-NSGA-III on welded beam design problem.....	143
FIGURE A.10:	Performance comparison between NSGA-II, NSGA-III and U-NSGA-III on pressure vessel design problem	144
FIGURE A.11:	Bi-objective multi-fold simulations (experiment 2) on unconstrained test problems	144
FIGURE A.12:	Bi-objective multi-fold simulations (experiment 2) on constrained test problems	145
FIGURE A.13:	Parallel Coordinate Plots (PCPs) of unconstrained normalized 10 objectives test problems	146
FIGURE A.14:	Parallel Coordinate Plots (PCPs) of constrained 10 objectives test problems	146
FIGURE A.15:	Final population reached on unconstrained normalized multi-objective problems	147
FIGURE A.16:	Final population reached on constrained multi-objective problems	148

LIST OF ALGORITHMS

ALGORITHM 1: Generation t of NSGA-III procedure.....	20
ALGORITHM 2: Niching based selection of U-NSGA-III.	39
ALGORITHM 3: Generation t of U-NSGA-III procedure.	40
ALGORITHM 4: Degenerated U-NSGA-III algorithm for single-objective problems.	41
ALGORITHM 5: KKTPM-based LS procedure.....	86
ALGORITHM 6: Phase 1.....	115
ALGORITHM 7: Phases 2 and 3	116
ALGORITHM 8: $\text{getBestWithinNiche}(All, D)$	117
ALGORITHM 9: $\text{fillUpPop}(All, N, \hat{P})$	118

Chapter 1

Introduction

Introducing Vector Evaluated Genetic Algorithms (VEGA) [1] was a turning point in the history of Evolutionary Computation (EA). This is the birthdate of Evolutionary Multi-objective Optimization (EMO). From this day on, the obligation of combining all objectives into one fitness function started to fade. The history of this relatively new field can be viewed from several perspectives. The one perspective that relates to our work here, is how EMO has developed over the years in terms of objective dimensionality i.e. the number of objectives. Roughly, the first fifteen years were focused on dealing with only two objectives [5]. Several algorithms with variable rates of success were proposed (see Chapter 2). Researchers then were concerned mainly about reaching and covering the whole set of trade-off solutions, or what is formally known as the "Pareto-front" ¹. Over the next ten years (roughly), generalization efforts started and flourished. It was clear that the best known algorithms in bi-objective optimization do not scale up to solve three or more objectives [6]. Consequently, several successful algorithms were proposed to deal

¹named after the Italian economist "Vilfredo Pareto"

with mainly three or four objectives. Only recently [4, 7–10], the term "Many-Objective Optimization" was coined to designate problems having a "large" number of objectives. In this context, the term "large" means *more than three*. Researchers were able to develop algorithms solving up to fifteen objectives (see Chapter 2). Today, the field is more active than ever. Hundreds of researchers from a wide variety of scientific disciplines have joined the field since its advent thirty years ago [1]. And it easy to tell how far the field has gone.

In spite of the number of researchers working in the field and the number of studies conducted, EMO is still facing many problems and challenges. In the next section we summarize two of the problems currently facing both researchers and practitioners developing and applying algorithms dealing generally with evolutionary optimization and specifically with more than one objective.

1.1 Problem Definition

Practitioners usually are obligated to follow one of two paths. The first path is to adapt the number of objectives they are dealing with to the algorithm/software package in-hand. This is usually done by combining several – may be conflicting – objectives into one. This approach is undesirable because unlike optimizing objectives separately, optimizing combined objectives is prone to loosing useful trade-off solutions, depending on the relation between these objectives and how they are combined. The second path is to use different algorithms, one for each level of objective dimensionality. This is an obvious inconvenient approach to avoid the problems caused by the first path. The source of this

problem is that most researchers develop their algorithms targeting one specific level of dimensionality, usually either single, bi- or multi-/many objectives.

On researchers' side, the ever-existing trade-off between convergence and diversity becomes more complicated as the number of objectives increases. They are usually conflicting. Hasty convergence ² may not give the algorithm enough opportunity to cover the whole set of trade-off solutions. Conversely, putting early emphasis on diversity can delay convergence. And without prior knowledge of problem landscape, it is difficult to tell in advance which of the two (convergence and diversity) should be the main concern of the optimizer. Because of this, most current algorithms maintain almost equal balance between the two. In a nutshell, we can say that the ultimate goal of this line of research is to create an algorithm that can dynamically balance its emphasis between convergence and diversity according to the problem.

1.2 Motivation

Solving the first problem involves carefully designing an algorithm that can adapt its operation to objective dimensionality automatically, without any hard-coded switches. This is important because although problems are usually categorized into three or four dimensional categories [4, 5, 11], the actual range of categories is more finely grained. In other words, being in the same many-objective category does not mean that an 8-objectives problem and a 15-objectives problem should be treated uniformly. Moreover, the deficiencies with existing algorithms to either scale up (to solve higher dimensional problems than initially anticipated) or down (to solve lower dimensional problems than

²premature convergence which results in reaching either an incomplete or a local Pareto front.

initially anticipated) makes it more difficult to rely on one of them to develop the targeted algorithm.

In addition to algorithmic motivations, there are other practical motivations for the first part of our study. We will discuss four of them here. First, In order to solve an optimization problem, it (the problem) must first be implemented (coded or expressed symbolically) within the optimizer (either a computer code or a commercial software). Often, this implementation process involves linking the optimizer to a third-party evaluation software such as a finite element or a computational fluid dynamics software or a network flow simulator etc. Secondly, in order to achieve better results, it is also recommended to customize the optimizer itself for the problem in hand [12, 13] by introducing new operators and/or modifying existing genetic operators utilizing the "heuristics" of the problem. For example, instead of starting an optimization run from a random initial population, a heuristically biased initial population is created. Such algorithmic modifications and customized initializations involve careful analysis, efforts and are certainly time-consuming. Thirdly, many multi- or many-objective optimization methods require the same problem to be solved for individual objectives one at a time for obtaining ideal and Nadir ³ points prior to solving the multi- or many-objective version of the problem. Often, such several lower-dimensional runs are executed to verify or gain confidence in the obtained higher-dimensional efficient front [14]. Fourthly, in design exploration problems, objectives and constraints are interchanged to get a better idea of the possible range of optimal solutions [15]. Taking these four considerations into account, assume that a distinct optimizer is

³Nadir point is the vector of the worst objective function values across all Pareto optimal solutions.

needed for each dimensional version of the original optimization problem. In such a situation, problem implementation, algorithmic modifications and custom initializations will need to be reimplemented for every optimizer, thereby making the overall process slow, tedious and error-prone. Solving a single objective version of a multi-objective problem will be more complicated as it will need a different optimizer. Finally, each version of a design exploration problem will need an optimizer that suites its dimensionality, which in turn depends on the specific combination of constraints used as objectives and vice versa. Instead, if one unified optimization algorithm capable of handling one to many objectives efficiently is available, then a one-time algorithmic modification based on heuristics, one-time implementation of the problem description and one-time linking with evaluation softwares would be more convenient for solving different dimensional versions of the original problem. This provides flexibility to users in moving back and forth between different objective-dimensions of the same original problem and also saves time, effort and most importantly makes the process less error-prone.

Developing a unified algorithm that can dynamically adapt to dimensionality is only one part of the story. We also need to make sure that when it comes to more than one objective, our state of the art algorithm is able to put more emphasis on either convergence, diversity or both. Varying the emphasis of the algorithm can capture difficult-to-attain solutions and/or reveal interesting aspects about the problem. It is very useful for a practitioner – with little or no information about the problem – to try to put emphasis on convergence at one time and on diversity at another time. And since it is very difficult to tell beforehand what kind of emphasis the current problem needs, it will be of great practical usefulness if the algorithm itself can dynamically sense the need for more convergence and/or diversity

based on the problem being solved.

1.3 Thesis Statement

This thesis develops a scalable unified algorithm that is able to automatically balance its focus between convergence and diversity. This algorithm is created through an efficient combination of EMO, Karush Kuhn Tucker (KKT) conditions and point-to-point single objective local search. Compared to state of the art algorithms including CMA-ES, NSGA-III and MOEA/D – among others – the algorithm is shown to have promising results over single-to-many objectives spectrum.

1.4 Contribution

A unified evolutionary optimization algorithm that can scale from one to many objectives. With more than one objective, the algorithm should be smart enough to put its resources where they are most needed by helping poorly converged solutions, covering efficient front gaps and expanding coverage whenever possible. Through this selective strategy the algorithm should be able to automatically balance its focus between convergence and diversity throughout the optimization run. Creating this algorithm enables one-time implementation of solution representation, operators, objectives and constraints formulations while maintaining an outstanding performance across several objective dimensions. The next subsections discuss the major parts of this study in brief.

1.4.1 Unified EMO for Single, Multiple and Many Objectives

In the first part of this dissertation and for the first time, we propose a unified evolutionary optimization algorithm (U-NSGA-III) [16] for solving all three classes of problems (single, multi and many), based on the recently-proposed, decomposition-based, elitist, guided, Non-dominated Sorting Genetic Algorithm (NSGA-III) originally developed to solve many-objectives problems. NSGA-III uses multiple pre-defined yet adaptable reference directions to maintain diversity among its solutions. With the intent of solving many-objective problems, the authors of NSGA-III restricted population size to be equal to the number of chosen reference directions (single-fold restriction). This restriction hinders the usage of NSGA-III with single-objective optimization problems, where, by definition, there is only one reference direction. U-NSGA-III is capable of adapting automatically to the dimensionality of the problem in hand through its niching-based selection operator. It automatically degenerates to an efficient equivalent population-based algorithm for each class. No extra tunable parameters are needed. Extensive simulations are performed on unconstrained and constrained test problems having single, two, multi and many-objectives and on two engineering optimization design problems. The performance of the unified approach is compared with a suitable population-based counterpart at each dimensional level. Results amply demonstrate the merit of our proposed unified approach, encourage its further application, and motivate similar studies for a richer understanding of the development of optimization algorithms.

1.4.2 Selection in NSGA-III

Because of the role played by selection in the first part of this study, we were urged to further investigate its effect on this type of algorithms (reference-based, population-based, evolutionary, multi-objective optimization algorithms) [17]. Unlike the first part of this study, here we ignore the single-fold restriction (described above). In other words we allow population size to increase freely beyond the pre-defined number of reference directions (multi-fold approach). Over a wide range of constrained, unconstrained, single, multi- and many-objective problems, we investigate the strengths and weaknesses of multi-fold NSGA-III compared to those of U-NSGA-III. The robustness of NSGA-III in each type of problems is also discussed. This study provides a more comprehensive evaluation of the original NSGA-III procedure, which seems to have a wider scope than what both the original study [4] and the first part of our study (U-NSGA-III) had foreseen.

1.4.3 Enhancing Convergence in EMO

In Chapter 5, we propose a multi-objective evolutionary algorithm with emphasis on convergence. Traditionally, evolutionary multi-criterion optimization (EMO) algorithms emphasize non-dominated (convergence) and less-crowded (diversity) solutions in a population iteratively until the population converges close to the Pareto-optimal set. During the search process, non-dominated solutions are differentiated only by their local crowding or contribution to hypervolume or using a similar other metric. Thus, during evolution and even at the final iteration, the true convergence behavior of each non-dominated solution from the true Pareto-optimal set is unknown. Recent studies [18, 19] have used Karush-Kuhn-Tucker (KKT) optimality conditions to develop a KKT Proximity Measure

(KKTPM) for estimating proximity of a solution from Pareto-optimal set for a multi-objective optimization problem. In this paper, we integrate KKTPM with NSGA-III to enhance its convergence properties towards the true Pareto-optimal front. Specifically, we use KKTPM to identify poorly converged non-dominated solutions in every generation and apply an achievement scalarizing function based local search procedure to improve their convergence. Assisted by KKTPM, the modified algorithm is designed in a way that maintains the total number of solution evaluations⁴ as low as possible while making use of local search where it is most needed. Simulations on both constrained and unconstrained multi- and many-objective optimization problems demonstrate that the hybrid algorithm significantly improves overall convergence properties. In addition, this study brings evolutionary optimization closer to mainstream optimization field and should motivate researchers to utilize KKTPM measure further within EMO and other numerical optimization algorithms.

1.4.4 Balancing Convergence and Diversity

In the final part of our study (Chapter 6), we propose B-NSGA-III, a multi-phase version of U-NSGA-III that adds a diversity preservation mechanism to the convergence enhancement mechanism proposed in Chapter 5. B-NSGA-III combines EMO, KKTPM and two types of local search operators into one algorithm. The algorithm is carefully designed so that all these elements serve their designated purposes cooperatively rather than competitively. The key behind cooperation is the alternating nature of the algorithm. B-NSGA-III alternates seamlessly among three phases according to the current situation

⁴ In a multiobjective optimization context, a Solution Evaluation (SE) refers to evaluating all the objectives of one solution. Like Function Evaluation (FE) in single objective optimization, an efficient multiobjective optimization algorithm should yield good performance with the minimum possible number of SEs.

faced by the algorithm. Thus, it adapts itself according to the problem it currently solves. Logical justifications and extensive simulations are presented to show the usefulness of the proposed multi-phase approach.

1.5 Summary

This chapter puts the rest of this study in its designated context and justifies its usefulness. Initially, we define our problem and explain the benefits both researchers and practitioners will gain from solving it. Then, we present our thesis statement to which everything in this study boils down. Finally, every contribution we make in this study is briefly discussed starting with creating a unified optimization algorithm and ending with achieving automatic balance between convergence and diversity.

The rest of this study is organized as follows. A brief history of EMO and a review of related work are presented in Chapter 2. Chapter 3 discusses our proposed unified approach (U-NSGA-III) in detail. Our selection study is presented in Chapter 4. In Chapter 5, we discuss our approach to enhance convergence and extend it to B-NSGA-III in Chapter 6. Finally, our conclusions and future work are discussed in Chapter 7.

Chapter 2

Related Work

In this chapter we provide a review of the most important efforts in EMO literature that relates to our study. Starting from a wide perspective, the discussion narrows down gradually towards our focus here. Finally, we discuss a set of challenges facing state of the art EMO, with a brief description of how our study handles each of them.

2.1 Introduction

Designing algorithms inspired by natural evolution started with John Holland's work in sixties and the seventies [20, 21]. In his book, Holland proposed the idea of Genetic Algorithm (GA). Several researchers followed his ideas, most notably David Goldberg who used GAs specifically in optimization applications [22–25]. Since that time researchers have been using GA's successfully in different optimization applications [25, 26]. Similar algorithms were proposed around the same time of Holland's work, including Evolution

Strategies [27, 28] and Genetic Programming [29, 30]. Survival-of-the-fittest is the principle upon which all these Evolutionary Algorithms (EAs) were designed. All of them are population-based, which means that optimization starts with a whole set of solutions (population). This population evolves from one generation to the next using various types of operators [25, 31–33]. These operators usually involve certain degrees of randomness. Solutions keeps interacting through available operators until some stopping criterion is met. Eventually, the fittest solution is returned. In their basic structure, EAs do not use gradients. The disadvantage of using EAs is that there is no guarantee or proof that the fittest solution will be the true global optimum. This outline opposes mathematical (classical) optimization point-to-point philosophy, where the optimization process moves from a single solution to the next in a deterministic way where gradients usually play an important role [11, 34]. In mathematical optimization, a point is proven to be a local optimum, given some conditions [35]. However, these conditions are rarely satisfied in real world problems. Real world problems can be discontinuous and/or multimodal. They can also be black box problems for which no mathematically formulated objective function is known. In such cases, no accurate gradient information can be obtained. For all these reasons, EAs outperform mathematical optimization techniques on many real world optimization problems.

2.2 Evolutionary Multiobjective Optimization (EMO)

EA researchers kept there focus on single objective optimization problem until Schaffer introduced his Vector Evaluated Genetic Algorithm (VEGA) in 1985 [1]. In his paper, Schaffer raises the following question, "How might survival of the fittest be implemented

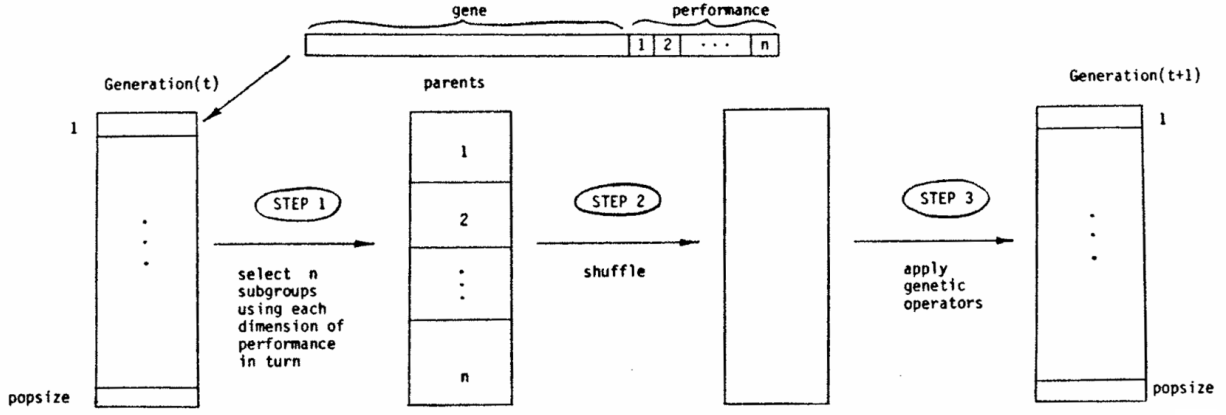


FIGURE 2.1: Schematic Description of VEGA (copied from [1])

when there is more than one way to be fit?" [1]. As an answer to this question, he proposed VEGA. In an M objectives problem, VEGA considers each objective to be – alone – the comparison criterion used in selection. Thus, M selection cycles are performed, each of which is responsible for contributing some of the members of the new population. This simple approach is outlined in Figure 2.1. The idea behind this simple approach is to maintain emphasis on all objectives throughout the entire optimization process. Although, the idea seems plausible, it can easily loose trade-off solutions for the sake of extreme solutions (i.e. those performing well only on a subset of the objectives). In order to avoid this negative effect, VEGA incorporated a penalization scheme based on the non-domination concept proposed by Pareto [36, 37]. Non-domination is a relation between two vectors. In a minimization context¹, a vector $X = (x_1, x_2 \dots x_m)$ is said to be non-dominated with respect to $Y = (y_1, y_2 \dots y_m) \iff x_i \leq y_i \forall i$ and $x_i < y_i$, for at least one i . In this case, Y is said to be dominated-by or inferior-to X . For a proper² multiobjective optimization problem, there will be a set of non-dominated solutions with respect to all

¹ Throughout all parts of this study, the default context is minimization unless otherwise stated.

² A proper multiobjective optimization problem is a problem where objectives are conflicting. In other words, there is no single solution that achieves the minimum (best) of all objectives simultaneously

other solutions in the feasible search space. Obviously, an EMO algorithm targets this set. In VEGA, dominated solutions are penalized and non-dominated solutions are rewarded. Since then the non-domination concept has become a very important factor in EMO.

Over the past two decades, many successful evolutionary multi-objective optimization (EMO) algorithms have been proposed. Among the first attempts of EMO methods are MOGA [38], NPGA [39] and NSGA [40], where researchers devised different possible ways of handling more than one objective by carefully balancing emphasis between convergence and diversity preservation. For achieving convergence to Pareto-optimal solutions, these algorithms continued to further incorporate the concept of Pareto domination [5]. MOGA was the first algorithm to introduce the idea of grouping solutions into different layers based on their non-dominated ranking. A true Pareto optimal solution will never be dominated, consequently it will always be ranked 1. In terms of diversity preservation, this family of algorithms continued to use several diversity-preservation methods studied in the context of single-objective evolutionary computation methods [23, 41, 42].

A second wave of algorithms followed. These algorithms incorporated elite-preservation concept which ensures the survival of non-dominated and well-diversified solutions from one generation to the next. Some of the most prominent studies of this wave are NSGA-II [2], SPEA [43], SPEA2 [44] and PAES [45] among others [5, 46–49].

Out of all these algorithms NSGA-II has been the most widely used algorithms in the field [50–54]. In NSGA-II, elitism is ensured by merging both parents and their offspring into one double-sized combined population, sorting them, then keeping the better half of the

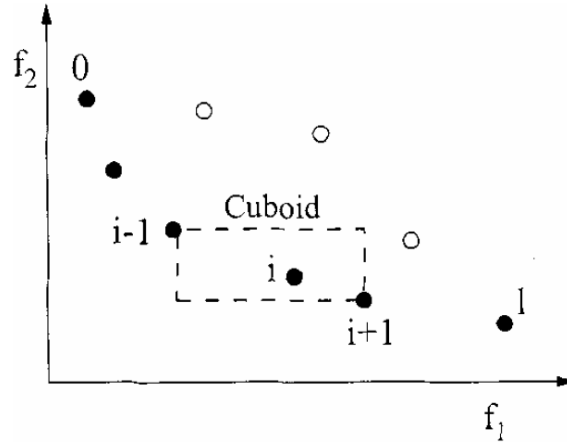


FIGURE 2.2: Crowding Distance in NSGA-II (taken from [2])

combined population. For diversity preservation, the authors of NSGA-II proposed an efficient approach where each member is assigned a crowding-distance value. This value is simply an approximation of the perimeter of the cuboid whose edges are those pairs of solutions surrounding the designated solution in each dimension as shown in Figure 2.2. The larger the crowding-distance value, the more important the solution is. This is because such an individual³ is considered the only representative of a less crowded portion of the objective space. By continuously emphasizing lower ranks then larger crowding distance values (in this specific order), NSGA-II achieves high levels of both convergence and diversity eventually. The whole NSGA-II procedure is shown in Figure 2.3. On the other hand, SPEA and SPEA2 maintains an explicit external population of well-converged and well-diversified solutions and use them for creating new populations. PAES performs a competition between a parent and its child to enforce elite-preservation. Second wave algorithms – in general – suffers their inability to scale up to more than two objectives.⁴

In response to the increasing need for algorithms capable of handling many objectives, a

³ The terms "solution", "individual" and "points" are used interchangeably throughout this study.

⁴ It is worth noting that some of the later ones were shown to solve up to 3 objectives as well.

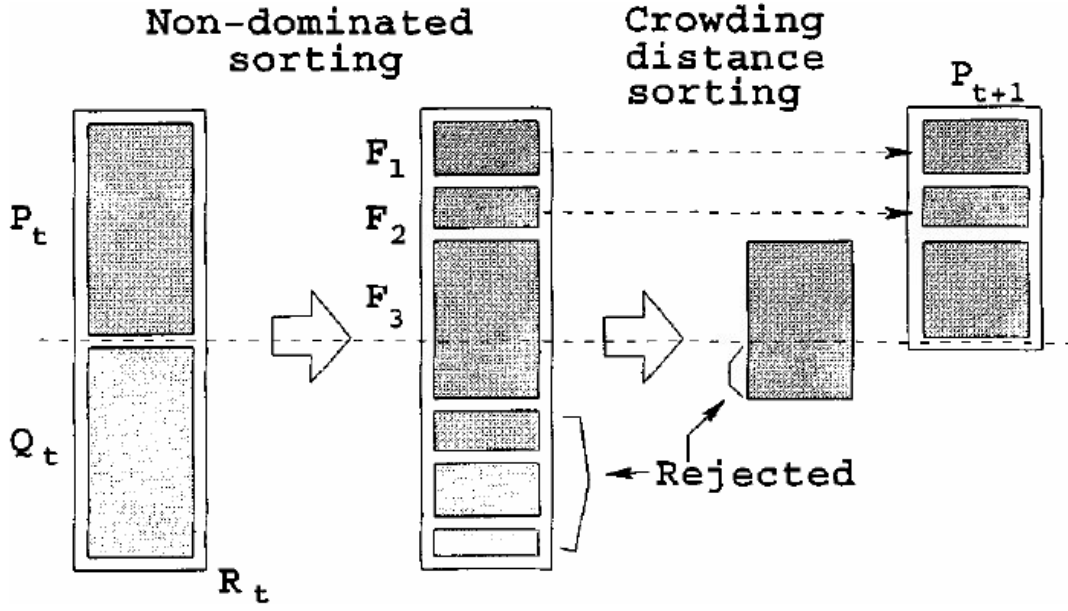


FIGURE 2.3: NSGA-II Algorithm (taken from [2])

third wave of EMO algorithms emerged in the last decade [3, 6, 55–66]. Zhang and Li’s MOEA/D [3] had the right ingredients for such an algorithm. They proposed a decomposition based method, guided by a set of predefined evenly distributed set of reference directions in the objective space. The original multiobjective optimization problem is decomposed into a set of single objective optimization subproblems, one problem for each reference direction. Each subproblem utilizes information from its neighbouring subproblems only. The authors proposed several approaches for formulating these subproblems. The most successful approach is the Penalty-based Boundary Intersection (PBI) approach shown in Figure 2.4 (notice that the original study was proposed in a maximization context). In this approach, for a given point x , objective vector $F(x)$, reference point z^* and reference direction λ , the weighted sum of the two distances d_1 and d_2 – in the objective space – is minimized. d_1 represents the Euclidean distance between z^* and the projection of $F(x)$ on λ , while d_2 represents the Euclidean distance between $F(x)$ and its projection

on λ . Other formulation approaches including simple weighted sum, Tchebycheff and non-penalized Boundary Intersection (BI) have also been discussed.

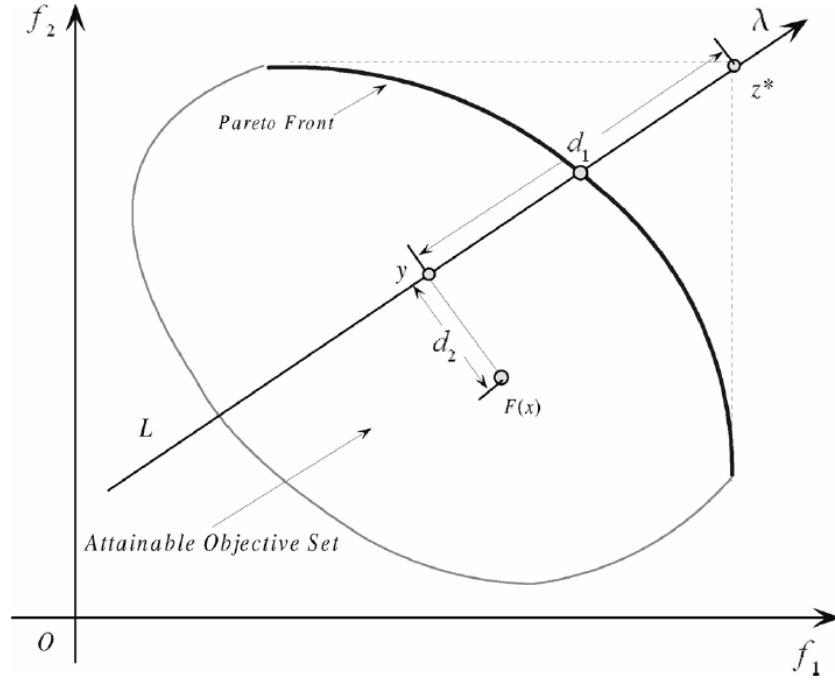


FIGURE 2.4: MOEA/D PBI (taken from [3])

Although their original study showed better results compared to NSGA-II and other algorithms up to 4 objectives, their algorithm have been used successfully elsewhere up to 10 objectives [4]. In 2013, Deb and Jain extended NSGA-II to handle many-objective optimization problems and proposed NSGA-III [4][7]. Their study included problems ranging from 3 to 15 objectives. NSGA-III is parameterless. It uses the same general framework of NSGA-II, but with a modified diversity-preserving operator that is based on a decomposition concept similar to that of MOEA/D. Although, the original study shows the outstanding performance of NSGA-III in the many objectives realm, the authors did not include neither bi-objective tests nor any comparison between NSGA-III and its predecessor NSGA-II. Since our study builds on top of NSGA-III, we devote the next

section to discuss its general philosophy and some of its details. For detailed explanations and extensive simulation results the reader is advised to consult [4, 7].

2.3 Introduction to NSGA-III

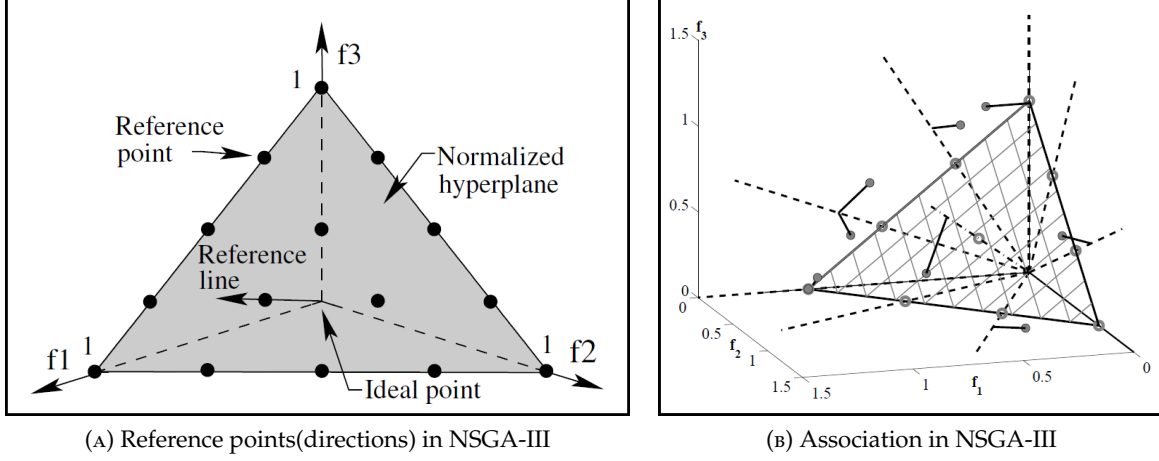


FIGURE 2.5: Niching in NSGA-III (taken from [4])

NSGA-III starts with a random population of size N and a set of H evenly-distributed, pre-specified M -dimensional *reference points*⁵ on a unit hyper-plane having a normal vector of Ones covering the entire R_+^M region. The hyper-plane is placed in a manner so that it intersects each objective axis at 1 (see Figure 2.5a ([4])). Das and Dennis's technique [67] is used to place $H = \binom{M+p-1}{p}$ reference points on the hyper-plane having $(p+1)$ points along each boundary. The population size N is chosen to almost equal to H , with the idea that for every reference point, one population member is expected to be found.

At generation t , the following operations are performed. An offspring population Q_t is created from the parent population P_t using usual recombination and mutation operators.

⁵ Although the original study used the notion of a "reference point", here we will mostly use the notion of a "reference directions", which is the vector connecting the ideal point to the "reference point". We found this notion more conceivable and avoids the confusion that may arise between the two terms "point" (an actual solution) and "reference point" (a hypothetical point in the objective space).

Since only one population member is expected to be found for each reference point, there is no need for any selection operation among feasible solutions in NSGA-III, as any selection operator will allow a competition to be set among different reference points. A combined population $R_t = P_t \cup Q_t$ is then formed, and the whole combined population is sorted into different non-domination levels, in the same way it is done in NSGA-II. Thereafter, solutions starting from the first non-dominated front are included in P_{t+1} (the next parent population) one front at a time. Typically, the algorithm will reach a front that has more individuals than the remaining slots in the next parent population i.e. a front that cannot be fully accommodated in P_{t+1} . Let's denote this last front as F_L . In such a case, a niche-preserving operator is used to select the subset of F_L that will be included in P_{t+1} . The niche-preserving operator works as follows. First, each population member of P_{t+1} and F_L is *normalized* using the current population spread so that all objective vectors and reference points have commensurate values. Thereafter, each member of P_{t+1} and F_L is associated with its closest reference direction (in terms of perpendicular distance). Then a careful *niching* strategy is employed to choose those F_L members that are associated with the least represented reference points in P_{t+1} . The niching strategy puts an emphasis on selecting a population member for as many supplied reference directions as possible. A population member associated with an under-represented or un-represented reference direction is immediately preferred. With a continuous stress on non-dominated individuals, the whole process is then expected to find one population member corresponding to each supplied reference point close to the Pareto-optimal front, provided the genetic variation operators (recombination and mutation) are capable of producing respective solutions. The use of a well-spread reference directions ensures a well-distributed set of trade-off

solutions at the end (see Figure 2.5b ([4])).

ALGORITHM 1 Generation t of NSGA-III procedure

Input: H structured reference points Z^s or supplied aspiration points Z^a , parent population P_t

Output: P_{t+1}

```

1:  $S_t = \emptyset, i = 1$ 
2:  $Q_t = \text{Recombination+Mutation}(P_t)$ 
3:  $R_t = P_t \cup Q_t$ 
4:  $(F_1, F_2, \dots) = \text{Non-dominated-sort}(R_t)$ 
5: repeat
6:    $S_t = S_t \cup F_i$  and  $i = i + 1$ 
7: until  $|S_t| \geq N$ 
8: Last front to be included:  $F_l = F_i$ 
9: if  $|S_t| = N$  then
10:   $P_{t+1} = S_t$ , break
11: else
12:   $P_{t+1} = \bigcup_{j=1}^{l-1} F_j$ 
13:  Points to be chosen from  $F_l$ :  $K = N - |P_{t+1}|$ 
14:  Normalize objectives and create reference set  $Z'$ :  $\text{Normalize}(\mathbf{f}^n, S_t, Z', Z^s, Z^a)$ 
15:  Associate each member  $\mathbf{s}$  of  $S_t$  with a reference point:  $[\pi(\mathbf{s}), d(\mathbf{s})] = \text{Associate}(S_t, Z')$   %
     $\pi(\mathbf{s})$ : closest reference point,  $d$ : distance between  $\mathbf{s}$  and  $\pi(\mathbf{s})$ 
16:  Compute niche count of reference point  $j \in Z'$ :  $\rho_j = \sum_{\mathbf{s} \in S_t/F_l} ((\pi(\mathbf{s}) = j) ? 1 : 0)$ 
17:  Choose  $K$  members one at a time from  $F_l$  to construct  $P_{t+1}$ :  $\text{Niching}(K, \rho_j, \pi, d, Z', F_l, P_{t+1})$ 
18: end if

```

The original NSGA-III study [4] have been demonstrated to work well from three to 15-objective DTLZ and other problems. A key aspect of NSGA-III is that it does not require any additional parameter. The method was also extended to handle constraints without introducing any new parameters [7]. That study has also introduced a computationally fast approach by which reference directions are adaptively updated on the fly based on the association status of each of them over a number of generations. The algorithm is outlined in Algorithm 1.

2.4 Convergence versus Diversity in EMO

Balancing convergence and diversity has been drawing researchers' interest since the first ever multiobjective optimization algorithm [1]. Most early studies gave higher priority to convergence over diversity [5]. Gradually, researchers started to realize that strictly following this specific order might be too restrictive [3]. In this section we will discuss a selected set of notable efforts showing the progression of research in this topic.

2.4.1 Towards a Better Balance

In 2001, Deb and Goel proposed Controlled Elitist NSGA-II [68]. Their algorithm used a geometric distribution with a user defined parameter r that caps the number of allowed individuals in each front, in an exponentially decreasing order. Their approach is intended to preserve diversity by keeping a larger number of fronts represented in the population at all times. They showed that their approach can use this diversity preservation scheme to enhance convergence as well. A few years later, Bosman and Thierens discussed several ways by which EMO algorithms performs exploitations and exploration of both proximity (convergence) and diversity [69]. In their study they stated that "the exploitation of diversity should not precede the exploitation of proximity". Yet they warned that delaying diversity preservation can result in finding only discontinuous sections of the Pareto front instead of showing the entire trade-off. They also suggested a parameter-based approach to control how much emphasis is put on diversity.

From a different perspective, the desired balance can be attained through variation operators. The study conducted by Tan et al. [70] explored this idea in the context of binary

chromosome representation. They proposed an Adaptive Variation Operator (AVO). AVO utilizes the chromosomal structure to tune crossover and mutation rates of each gene independently. Genes representing most significant bits are assigned higher rates in the beginning of optimization to encourage exploration. These rates decrease (either linearly or non-linearly) as optimization proceeds. The opposite is applied to genes representing the least significant bits, in order to encourage exploitation at the final stages of optimization. Thus, emphasizing diversity (a consequence of exploration) then moving gradually towards emphasizing convergence (a consequence of exploitation). They also combined crossover and mutation in a way intended to prevent mutation from disrupting the flow of information created by crossover. Despite the restricted context of this study, it represents the converse of the commonly adopted convergence-first idea at the time. However, It is important to emphasis that their results are based on the assumption that a chromosome is the direct binary representation of a number. Consequently, their conclusion cannot be extended, even over binary chromosomes in general.

Due to the recent success of many-objective optimization algorithms (> 3 objectives) [3, 4, 7, 16], balancing convergence and diversity becomes even more challenging than before. As the number of objectives grows, the percentage of non-dominated solutions increases significantly. Consequently, one of the most widely used converging forces – non-domination – becomes ineffective. Although this effect can be generally considered a disadvantage, it can be beneficial in the realm of many objectives. As the dimensionality of the objective space increases, EMO algorithms become more prone to loosing parts of the Pareto front due to premature convergence. In such cases, maintaining mild convergence pressure allows for more exploration, which in turn can result in better diversity. This is

the reason behind the reduced selection pressure adopted in NSGA-III [4].

In this context, Ke Li et al. proposed an interesting extension to MOEA/D [71]. Their approach treats subproblems and solutions as two different types of agents. Each subproblem creates a preference list of all available solutions sorted by their performance in the employed scalarization function (which is a metric of convergence in this specific subproblem). On the other hand, each solution creates a preference list of all available subproblems sorted by the perpendicular distance between the solution itself and each of them (which signifies diversity). After each agent – from both sides – creates its own preference list, a stable matching algorithm [72] dictates the final subproblem-solution associations. Those selected solutions move to the next generation.

Researchers continued to experiment with MOEA/D in the last few years. Wang et al. proposed a modification of its replacement strategy (Global replacement) [73]. The original version of MOEA/D simply assumes that a solution coming out of a specific subproblem can only replace another solution in the vicinity of this subproblem. Their approach on the other hand looks for the subproblem at which the new solution fits best, and performs replacement in the vicinity of this subproblem instead of the original one, hence the name "Global replacement". They showed how this simple modification performs better on a set of test problems. Another recent study by Yuan et al. [74] proposed a modified version of MOEA/D that achieves better balance. In their study they use a slightly relaxed mating restriction (controlled by a probability parameter δ). Their approach – MOEA/D-DU – utilizes the perpendicular distances between solutions and directions (d_2) for enhanced diversity. Unlike the Penalty Boundary Based Intersection MOEA/D (MOEA/D-PBI) they are not incorporating d_2 in the scalarization function itself. They are still using Tchebycheff

scalarization function. Instead, they compare each new solution (from subproblem F_j) with a non-decreasing sorted list of the K closest solutions to F_j (sorted by d_2). If the new solution y has a better Tchebycheff value than solution x_k , y replaces x_k immediately. It is worth noting that MOEA/D-DU is a steady state algorithm. The study also included a similar extension of EFR [75].

Most of these studies do not propose truly *automatic* balancing approaches. We can generally classify them into two categories. The first category follows a predefined preference scheme to achieve the desired balance, either by focusing on convergence then diversity, or doing the opposite! Although following either way can show some merit on a selected set of problems, none of the two approaches can be considered appropriate in general. In addition, many problems would not fit in any of these two extremes. A parallel emphasis would be more robust and predictable over the whole spectrum of optimization problems. This parallel approach is adopted by the second category, however, this category uses an additional user-defined parameter to indicate the relative effort put to either convergence or diversity. Now it is the user's responsibility to find the right value for this parameter, which is a very challenging task given a new – possibly black-box – optimization problem. Thus, none of these approaches can be considered truly *automatic*.

One of the contributions of this study is the infinite seamless alternation of phases it follows, in order to reach the desired balance without adding explicit preferential parameters. In the remaining part of this chapter we study some key components that enabled us to reach this balance.

2.4.2 Optimality in EMO

Karush-Kuhn-Tucker (KKT) conditions are one of the early and most widely used forms of optimality conditions [11, 76–78]. KKT conditions generalize the method of Lagrange multipliers previously applicable only to handle equality constraints. They are necessity conditions that each optimal point must satisfy. The opposite is not true, however. A KKT point is not necessarily optimal. Despite this fact many researchers and even popular softwares use some form of KKT conditions to check the optimality of their solutions. Thus, it would be more accurate if we said that these studies/software search for KKT points, instead of actual optimal points. To ensure optimality another set of more involved conditions (sufficiency conditions) should be used. But, sufficiency conditions are more difficult to apply in real world problems. For example, KKT conditions are sufficient (ensure global optimality) if the problem satisfies moderate convexity assumptions, a rare case in practice.

Recent studies [79, 80] have clearly revealed one aspect: KKT optimality conditions are "singular" properties, applicable only at a KKT point (including an optimal point). The extent of violation of KKT optimality conditions cannot be construed in any way to predict a point's closeness to a KKT point. In other words, KKT conditions do not provide any clue regarding how far an arbitrary solution can be from being a true KKT point. Thus, in their original form, KKT conditions cannot be used as a convergence metric. To tackle this shortcoming, Dutta et al. [79] proposed a KKT proximity metric (KKTPM) computation procedure which behaves monotonically as points get closer to a KKT point for single-objective optimization problems. In 2015, Deb et al. extended this work for multi-objective

optimization problems [18] and showed that the respective KKTPM value for any point has a high correlation to the distance of the point from the efficient frontier (in the objective space). However, one drawback of the approach is that computing KKTPM value for a single point requires solving an auxiliary optimization problem with parabolic constraints, for which the number of variables is two more than the total number of constraints in the original problem. A later study [19] proposed an approximation procedure to compute KKTPM. In [19], Deb et al. showed that computing approximate KKTPM does not require solving a full optimization problem anymore, instead, a linear system of equations is solved to find the approximate value. This procedure opened the door wide for using KKTPM in various contexts.

The ability to efficiently compute a reliable value of KKTPM allows for its use beyond simply ascertaining termination of an optimization run – which was the initial motivation behind proposing KKTPM. In an EMO algorithm, non-dominated solutions are emphasized more than dominated solutions and when solutions from an identical non-dominated level are compared, the one having less crowding in its vicinity (in the objective space) is emphasized. Some algorithms compute a crowding measure [2, 44] to each solution for this purpose, while others compute contribution of each solution to an overall performance measure, such as incremental hypervolume [81]. However, none of these diversity indicators nor any other proposed measures are able to rank non-dominated solutions in terms of their convergence from the true Pareto-optimal set. It is unlikely that all non-dominated solutions are *equally* convergent to the true Pareto-optimal set at any generation, and niching properties alone cannot show. Moreover, there can be certain "spurious" population members which remain non-dominated in the population, but

are not close to true Pareto-optimal set. If only some critical Pareto-optimal points are discovered, these spurious solutions will be termed dominated in the population. None of the existing operators allow us to identify such poorly converged yet non-dominated population members. The discovery and fast computation of KKTPM fills the gap and allows us to identify poorly converged population members, which can then be operated on by means of a Local Search (LS) procedure to enhance their convergence properties. In this sense, our proposed study is a hybrid that fits mathematical optimization elements (KKT concepts and point-to-point LS procedures) into a larger EMO framework. Section 2.4.3 discusses the most important attempts made by researchers to utilize LS in an EMO context.

2.4.3 Local Search

In a multi-objective optimization context, *local search* (LS) refers to optimizing an aggregate (combined) form of all the original objective functions. So, to perform an LS that targets one Pareto point, you/the algorithm needs to (i) pick a starting point, and (ii) specify a search direction (i) pick a starting point, and (ii) specify a search direction (in the objective space).

- **Pick a Starting Point**

Although, it may be straightforward to prefer non-dominated over dominated solutions, it is not clear which non-dominated ones! If two points are non-dominated with respect to each other they are non-comparable by definition (this is the whole idea behind the non-domination concept). In such a case, Picking your starting point is not an easy task. Choosing one non-dominated solution at random can be

a strategy. However, it is obviously prone to pick points that are already close to the true front on the expense of other points that are still struggling although still non-dominated! Without further information, any adopted selection scheme will be ad-hoc and any claims about its broad applicability can be easily refuted.

- **Pick a Search Direction**

Choosing a search direction needs careful attention as well. Fortunately, many scalarization strategies exist in the MCDM literature [11, 34, 82] and a suitable one can be chosen for this purpose. There also exist derivative-based methods [83, 84]. It worth noting that this step has direct impact on the diversity of the final efficient set.

Several studies used LS in EMO. Ishibuchi et al. started this interesting combination in IM-MOGLS [85]. Their approach uses a simple weighted-sum aggregate (scalarization) function to combine all objectives. IM-MOGLS then starts an LS from each point in the population. Because of the possibly large amount of function evaluations consumed by each LS, another study proposed a modification where LS is applied to a selected set of points only [86]. After the proposal of NSGA-II, Ishibuchi and Narukawa proposed an NSGA-II extension that uses LS to solve multiobjective 0/1 knapsack problems [87]. Other researchers followed the same path by adding LS to already existing EMO algorithms. Knowles and Corne proposed a similar extension to PAES [88].

Harada et al. proposed the notion of a Pareto descent direction, which is a direction "to which no other descent directions are superior in improving all objective functions" [89].

Their algorithm, Pareto Descent Method (PDM), solves a Linear Programming (LP) problem to get each of these directions. However, PDM is hardly considered an evolutionary algorithm. It does not involve any genetic operators either for interaction among individuals (i.e. recombination) or for randomly modifying each individual independently (i.e. mutation). PDM is rather an Evolutionary Strategy (ES)-like algorithm where mutation is replaced by an LS in the Pareto descent direction of each solution. Later, Bosman proposed and tested several multiobjective gradient-based algorithms in [90]. Starting at some solution, he also provided an analytical representation of the directions in which objective values can not deteriorate (they either improve or remain constant). And although Bosman's work considers EMO algorithms for hybridization, it can be extended beyond the evolutionary domain.

The reader can notice that different studies used different ways to combine objectives i.e. scalarization methods. [82, 91]. Eventually, all these studies conduct LS in some directions. However these directions may harm diversity if they are calculated based on potentially decreasing objective values only. In addition, some scalarization functions may cause further complications in specific situations. For example, using a weighted sum approach yields the algorithm unable to attain non-convex sections of the Pareto front. For this reason, we generally recommend using Achievement Scalarization Function (ASF) as a general way of combining objectives, as it enables the algorithm to reach any point – theoretically – regardless of the convexity of the region at which this point lies on the front. In addition, given a reference directions based algorithm like NSGA-III, ASF can use those readily available directions. On the other hand, we have also noticed that using ASF can significantly distort the contour lines of the aggregate function if the direction (weight

vector) is too flat or steep. Consequently, optimizing ASF exhibits poor performance when used to reach objective-wise extreme points. This means that ASF is not a universal formulation that can be used unconditionally. Now, the reader can obviously see that before thinking about the single objective optimization algorithm that LS is going to use, we need to pay careful attention to the formulation of the aggregate optimization function itself. How the objectives are combined can significantly affect results either positively or negatively according to the situation.

Throughout this study, we are not concerned with the specific single objective optimization algorithm used by the single objective LS optimizers. We are more concerned with the formulation of the aggregate function itself, and how LS can be employed within the course of a bigger algorithm, in order to serve the ultimate purpose of automatically balancing convergence and diversity. One of the contributions of this study is using different formulations according to the current state/phase of the optimization process.

To reduce the overall computational burden, another algorithmic issue is whether to hybridize EMO and local search sequentially or concurrently [92]. If the hybridization is sequential, meaning that LS is to be performed after EMO is completed, the distribution of computational efforts between EMO and local search becomes crucial to the success of the overall hybrid approach. On the other extreme, a pure concurrent hybridization where every population member is used to start a local search (during the course of the larger optimization algorithm), will waste FEs significantly. Other intermediate schemes are possible, but they will also require a fine balance between LS and EMO on a case to case basis.

2.5 Road Map

The ultimate goal of this study, is to explore the possibility of creating a new algorithm that can efficiently move up and down the scale of objective dimensionality, while automatically balancing emphasis between convergence and diversity. In a systematic way, we first extend NSGA-III so that it can deal successfully with single and bi-objective optimization problems, and at the same time maintain its outstanding performance with many objectives. Then we combine NSGA-III with KKTPM and local search to create an algorithm that is able to identify ill-converged solutions and improve them with little sacrifice on diversity. Then we introduce another type of LS that can significantly enhance diversity. Finally, we try to combine the two techniques in a dynamic multi-phase procedure that can adaptively balance its focus based on the problem in hand without any prior knowledge of the problem or user predefined preferential parameters. Figure 2.6 shows the road map connecting all parts of this study.

2.6 Summary

In this chapter, we went through the most notable studies in the field. A brief overview of the history of evolutionary algorithms led to a detailed discussion of researchers' efforts in developing multi- and many-objective optimization algorithms, trying to balance convergence and diversity, formulating optimality metrics and the role local search played in previous studies.

The following chapters discuss how we address these challenges discussed in Chapters 1 and 2. Simulation results are conducted to verify our conclusions in each chapter

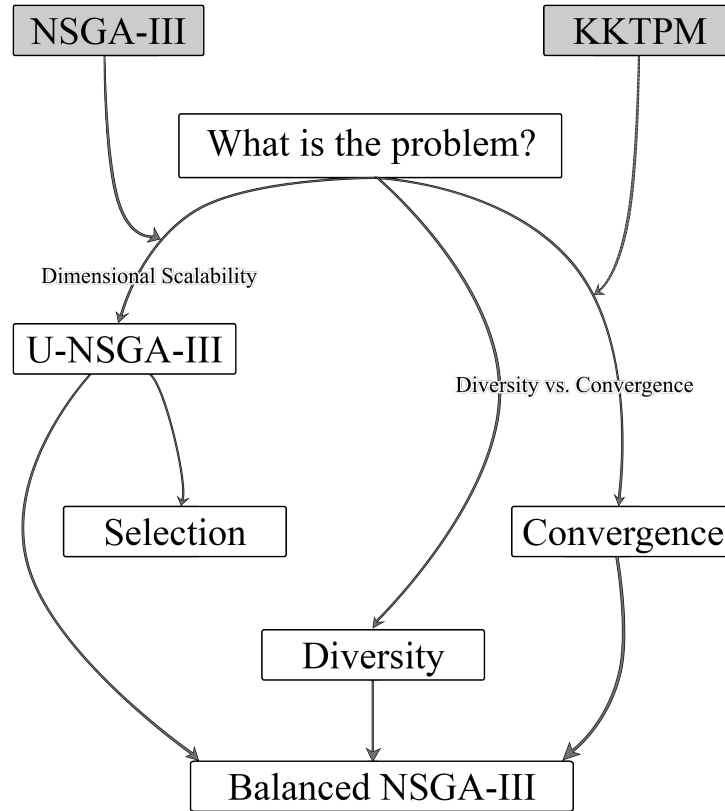


FIGURE 2.6: Road Map

separately. Each chapter starts with a detailed description of the algorithm and/or the idea and the logic behind it, followed by a set of extensive experiments and results. Finally a brief summary concludes. The final chapter discusses potential future extensions.

Chapter 3

A Unified Evolutionary Optimization Procedure for Single, Multiple, and Many Objectives

In this part of the study we propose our first contribution, U-NSGA-III. Initially, we discuss the operation of NSGA-III across different dimensions in Section 3.1. Then in Section 3.2, we discuss the details of our proposed algorithm and explain how it automatically degenerates to an efficient single, multi- or many-objective optimization algorithm according to the problem. Simulation results on a variety of single, multi- and many-objective problems, both constrained and unconstrained are presented using U-NSGA-III and compared to a real-parameter genetic algorithm (EliteRGA), CMA-ES [93], NSGA-II and NSGA-III in Section 3.3. Finally, conclusions are drawn in Section 3.4.¹

¹For better readability, additional figures and tables are moved to the appendix. A leading ‘A’ in some labels signifies such a figure or a table.

3.1 NSGA-III for Single and Multi-objective Problems

NSGA-III was primarily proposed to solve many-objective optimization problems having three or more objectives. And although the authors of the original study demonstrated clearly how NSGA-III deals successfully with this category of problems [4, 7], they did not include any single or bi-objective problems in the original study. Here, we discuss the potential of using NSGA-III in bi-objective problems and then highlight its difficulties in scaling down to solve single-objective optimization problems.

The differences in working principles of NSGA-II and NSGA-III on two-objective problems are outlined below:

1. Although both NSGA-II and NSGA-III give preference to feasible over infeasible and less violating over more violating solutions during the selection phase, NSGA-III's selection operator does not employ any preferential criteria when comparing two feasible solutions. In such a case NSGA-III picks one of the feasible solutions randomly. On the other hand, NSGA-II's selection operator compares feasible solutions using non-dominated ranking then crowding distance (if they have the same rank).
2. NSGA-III uses a set of reference directions to maintain diversity among solutions. This technique can scale easily up to higher objective dimensions. On the contrary, NSGA-II uses a more adaptive yet non-scalable scheme, through its crowding distance operator for the same purpose, as illustrated in Figure 3.1.

If NSGA-III is compared to NSGA-II with the same population size, the former will exhibit milder selection pressure. The rationale behind decreasing selection pressure is

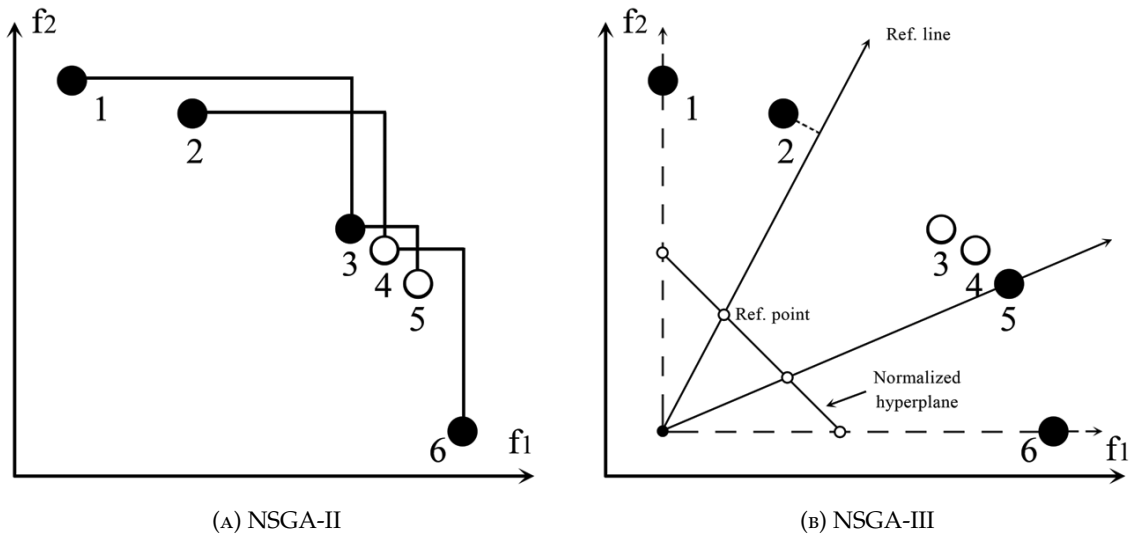


FIGURE 3.1: Working principles of NSGA-II and NSGA-III.

that – typically – each population member in NSGA-III becomes associated with a unique reference direction and becomes consequently too important to be compared to any other individual. The only selection pressure (among feasible) in NSGA-III comes from the non-dominated sorting phase.

Let us now discuss how NSGA-III would work on a single-objective optimization problem. In single-objective optimization, the domination concept degenerates to fitness superiority – a domination check between two solutions chooses the one having better objective value. Consequently, only one solution would occupy each non-dominated front in a single-objective problem². Thus, it is expected to have N fronts in a population of size N . These characteristics of single-objective problems affect the workings of NSGA-III in the following manner:

1. First, in NSGA-III, there will be only one reference direction (the real line) to which all individuals will be associated. Abiding by the single-fold restriction imposed

² Unless identical solutions exist.

by NSGA-III authors, the algorithm should use a population of size 1! For obvious practical considerations ³, we use 4 individuals instead, for all single-objective optimization problems. For all practical purposes, this population size is too small and prohibitive for NSGA-III's recombination operator to find useful offspring solutions. This is a major issue in developing a unified algorithm that will seamlessly work for many as well as single-objective problems.

2. Moreover, since no explicit selection operator is used among feasible solutions, the algorithm will pick a random solution for its recombination and mutation operators. In other words, the only selection pressure comes from the elite-preserving operation ⁴ used to choose P_{t+1} (next parent population) from a combination of P_t (current parent population) and Q_t (current offspring population). This is another major issue, which needs to be addressed while developing a unified approach.
3. Niching in NSGA-III becomes defunct when applied to single-objective problems, as there is the concept of perpendicular distance of an individual (in the objective space) from the corresponding reference direction is not useful anymore. All individuals fall on the real line, providing an identical perpendicular distance of zero.
4. NSGA-III's Normalization also becomes a defunct operation for the same above reason.

It is now clear that a straightforward application of the original NSGA-III to single-objective optimization problems will result in an extremely small population size and a random selection process, neither of which is recommended for a successful evolutionary

³ Evolutionary operators need more than one individual to conduct recombination

⁴ Non-dominated sorting ensures elite preservation in NSGA-III

optimization algorithm. On the other hand, niching and normalization operators are essential to the success of NSGA-III in multi and many-objective optimization problem. So, simply removing them is out of question. Hence, a modification of NSGA-III is needed so that the resulting unified approach becomes efficient for single to many-objective problems by making the niching and normalization operators automatically defunct for single-objective problems and active for multi and many-objective problems.

3.2 Proposed Unified Approach: U-NSGA-III

The above discussion suggests that overcoming the difficulties in scaling down to bi- and single-objective problems – mentioned above – require certain changes in the algorithm. But we need to modify NSGA-III in such a way so that the changes do not affect its working on three and more objective problems.

One way to alleviate these difficulties is to use a population size N that is larger than the number of reference points (H) and introduce a selection operator. Thus, unlike NSGA-III, N and H will now be independent parameters such that $N \geq H$. Although this seems to introduce an additional parameter to our proposed U-NSGA-III, it does not. H is the desired number of optimal solutions expected at the end of a simulation run, and hence it is not a parameter that needs to be tuned for U-NSGA-III to work well on different problems, it is rather a decision making preference. In Chapter 4 we show in detail that this change alone is not sufficient as it leaves most solutions wandering randomly in the search space without convergence force to drag them towards the Pareto front. For single-objective problems, we have to do so, because H is always 1 and N must be independent of H to move up freely. For bi-objective problems, keeping the single-fold restriction or

moving to multi-folds is a matter of choice though. If the multi-fold approach is adopted, H can be only a handful of solutions (such as 10 or 20) for the decision-makers to consider, while the population size N can be much larger, such as 100 or 200. The population size consideration mainly comes from the complexity of the problem and the desire for an adequate sample size that enables genetic operators (essentially the recombination operator) to work well. In this case, although N different Pareto-optimal solutions could be present in the final population, the H specific Pareto-optimal solutions, each closest to a unique reference direction will be the outcome of the U-NSGA-III algorithm and will be presented to the decision-maker for choosing a single preferred solution. For three or more objective problems, since the number of specified reference directions (H) can already be quite high (due to the increase in H with M according to Das and Dennis's approach [67]), we recommend keeping N almost equal to H .

Let us now discuss the algorithmic implications of introducing more population members than H when solving single and bi-objective optimization problems. It is now expected that for each reference direction, there will be more than one population member associated. We can use this fact to introduce a niching-based-selection operator that adapts selection pressure according to the problem in hand. We add a niching-based tournament selection operator as follows. If the two solutions being compared come from two different niches (associated to different reference directions), one of them is chosen at random, thereby preserving multiple niches in the population. Otherwise (if both are associated to the same reference direction), the solution coming from a better non-dominated front is chosen. Finally, if both solutions belong to the same niche and same non-dominated front, the one closer to the reference direction is chosen. It is important to emphasize that this operator

is applied when comparing feasible solutions. It is now obvious that unlike NSGA-III, U-NSGA-III does not consider all feasible solutions equal. Instead, it provides preferential treatment in the way we discussed. The pseudo-code shown in Algorithm 2 shows how our nichingbased selection operator compares two feasible parent solutions (p_1 and p_2) and pick a winner (p_s).

ALGORITHM 2 Niching based selection of U-NSGA-III.

Input: Two parents: p_1 and p_2
Output: Selected individual, p_s

```

1: if  $\pi(p_1) = \pi(p_2)$  then
2:   if  $p_1.rank < p_2.rank$  then
3:      $p_s = p_1$ 
4:   else
5:     if  $p_2.rank < p_1.rank$  then
6:        $p_s = p_2$ 
7:     else
8:       if  $d_{\perp}(p_1) < d_{\perp}(p_2)$  then
9:          $p_s = p_1$ 
10:      else
11:         $p_s = p_2$ 
12:      end if
13:    end if
14:  end if
15: else
16:    $p_s = randomPick(p_1, p_2)$ 
17: end if
```

Figure 3.2 shows graphically the new operator works in U-NSGA-III. In the case depicted, point **A** is the only representative of its niche (reference direction). So, if **A** goes through selection with any other point, one will be picked randomly. On the other hand since **B**, **C** and **D** belong to the same niche, they are comparable. **B** will be preferred over either **C** or **D** because it dominates both of them. If **C** and **D** are compared, **D** will be selected, because although both of them are non-dominated with respect to each other, **D** is closer to the designated reference direction. On the other hand, If at least one of them is infeasible, the traditional NSGA-III selection is used. This operation can be repeated $N/2$ times

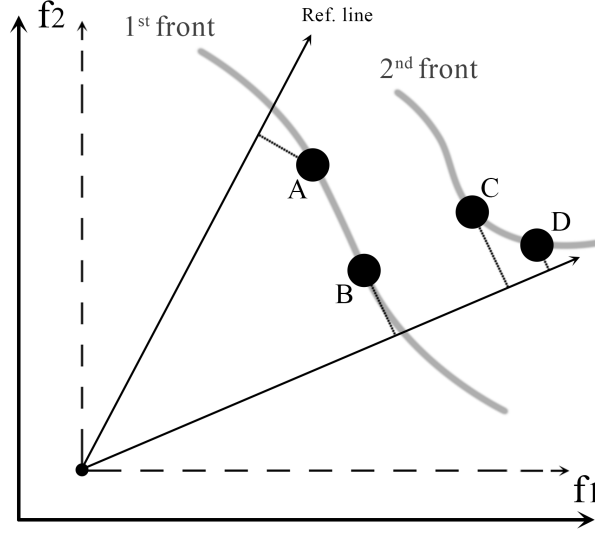


FIGURE 3.2: Selection in U-NSGA-III.

systematically by using two consecutive population members of the parent population P_t to choose $N/2$ parents. The procedure can be repeated one more time by shuffling population P_t to obtain another set of $N/2$ parents. These two chosen parent sets can be combined to form the complete mating pool P'_t of size N in the NichingBasedSelection(P_t) procedure. The mating pool P'_t can then be used to create the offspring population Q_t applying usual recombination and mutation operators. Thus, the complete U-NSGA-III procedure can be achieved by simply replacing line 2 in Algorithm 1 with the two lines shown in Algorithm 3.

ALGORITHM 3 Generation t of U-NSGA-III procedure.

Input: H structured reference points Z^s or supplied aspiration points Z^a , parent population P_t

Output: P_{t+1}

- $\vdots$ % Identical to Algorithm 1 (Line 1)
 - 2: $P'_t = \text{NichingBasedSelection}(P_t)$
 - 3: $Q_t = \text{Recombination+Mutation}(P'_t)$
 - $\vdots$ % Identical to Algorithm 1 (Lines 3 to 18)
-

For single-objective problems, the niched selection operator degenerates to a usual binary tournament selection operator for which the solution having a better objective value

becomes the winner. Next we discuss the behavior of U-NSGA-III with single, multi- and many-objective optimization problems.

3.2.1 U-NSGA-III for Single-objective Problems

The pseudo-code of the resulting U-NSGA-III (at $M = 1$) is presented in Algorithm 4. It is interesting to note that Das and Dennis's [67] strategy results in $\binom{1+p-1}{p}$ or one single reference point and is independent of the value of p . The resulting U-NSGA-III is a generational evolutionary algorithm that uses (i) a binary tournament selection, (ii) recombination and mutation operators, and (iii) an elite-preserving operator. Thus, our proposed U-NSGA-III is similar to other generational EAs, such as elite-preserving real-coded genetic algorithm [94] or the $(\mu/\rho + \lambda)$ evolution strategy [95], where $\mu = \lambda = N$ and $\rho = 2$.

ALGORITHM 4 Degenerated U-NSGA-III algorithm for single-objective problems.

Input: *Single-objective problem*

Output: Best solution found, p_{best}

```

1:  $P = initialize()$ 
2: while termination condition do
3:    $Q = \phi$ 
4:   while  $|Q| < |P|$  do
5:      $p_1 = tournamentSelect(P)$ 
6:      $p_2 = tournamentSelect(P)$ 
7:      $(c_1, c_2) = recombination(p_1, p_2)$ 
8:      $c_1 = mutate(c_1)$ 
9:      $c_2 = mutate(c_2)$ 
10:     $Q \cup \{c_1, c_2\}$ 
11:   end while
12:    $P = best(P \cup Q)$ 
13: end while
14:  $p_{best} = best(P)$ 

```

3.2.2 U-NSGA-III for Multi-objective Problems

For two or three-objective problems, in case N is chosen to be greater than H , U-NSGA-III is expected to have multiple population members for each reference direction. For multi-objective problems having two or three objectives, the non-dominated sorting will, in general, divide the population into multiple non-dominated fronts. The proposed niching based tournament selection operator of U-NSGA-III emphasizes (i) non-dominated solutions over dominated solutions and (ii) solutions closer to reference directions over other non-dominated but distant solutions from the reference directions. The rest of the U-NSGA-III algorithm works the same way as NSGA-II would work on multi-objective problems. All members of the final population are expected to be non-dominated to each other. The distribution of additional $(N - H)$ population members need not have a good diversity among them. Only H population members closest to each H reference directions are expected to be well distributed. This is not a problem, because the user is interested in reporting only H solutions at the end (implied by the value of H specified in the beginning). The additional $(N - H)$ points although not reported are expected to help throughout optimization.

As the factor N/H goes down towards 1, each niche will have approximately 1 solution. In these cases, our niching based selection operator tends to behave like the default NSGA-III selection operator. In either case ($N/H = 1$ or $N/H > 1$), U-NSGA-III provides the adequate selection pressure.

3.2.3 U-NSGA-III for Many-objective Problems

For many-objective optimization problems, most population members are expected to be non-dominated to each other. Hence, the niched tournament selection operator degenerates in choosing the closer of the two parent solutions with respect to their associated reference direction, when both parent solutions lie on the same niche. When N/H is much greater than one, this allows for an additional filtering that keeps solutions closer to reference directions to undergo subsequent mating operations. This – in general – is a good behavior, particularly when there are multiple population members available around a specific reference direction. But due to the explosion of reference directions generated by Das and Dennis approach in higher dimensions, U-NSGA-III with N greater than H may end up with a too large population. For this reason we recommend to have $N/H \approx 1$ in many-objective problems. This will yield the niching based selection operator defunct and the algorithm degenerates to the original NSGA-III.

The above properties of U-NSGA-III suggests a possible way to construct a single unified optimization algorithm that automatically degenerates to efficient optimization algorithms for single, multi- and many-objective optimization problems simply by the specification of the number of objectives presented in the problem description. The number of desired optimal points H is dictated by the number of points desired along each objective axis. The population size parameter N is detached from H and the user is free to provide any value greater or equal to H . U-NSGA-III is capable of handling constraints. The proposed niching based selection operator requires $O(N)$ computations. As discussed in NSGA-III study, the rest of computations is bounded by the maximum of $O(N^2 \log^{M-2} N)$ or $O(MN^2)$.

Thus, $M > \log^{M-2} N$, the generation-wise complexity of the overall U-NSGA-III procedure is $O(MN^2)$, which is similar to those of NSGA-II and NSGA-III.

3.3 Results

In the following sections, we present simulation results of U-NSGA-III applied to a variety of constrained and unconstrained single, multi- and many-objective problems. Two real-world engineering problems are also solved using the proposed U-NSGA-III algorithm. The stopping criterion is set according to the type of experiment being conducted. When the stopping criterion is a fixed maximum number of generations, this number is chosen according to the difficulty ⁵ of the problem. In most situations, more-than-required ⁶ number of generations is used to show that all algorithms were given enough time to reach their best performance. It is worth noting that due to the high complexity of calculating hypervolume, all hypervolume values in this study are calculated using the efficient implementation provided by JMetal optimization software package [96].

3.3.1 Single-objective Problems

First, we present the results of U-NSGA-III on standard single-objective optimization problems.

⁵The term "difficulty" when used with single-objective problems in this study refers to multi-modality. Finding the global optimum of a multi-modal problem needs a large population size and/or number of generations compared to a uni-modal problem.

⁶Empirically determined.

3.3.1.1 Unconstrained Problems

For the unconstrained case, we chose six single-objective test problems as our test-bed, namely, Ellipsoidal, Rosenbrock's, Zakharov's, Schwefel's, Ackley's and Rastrigin's. The problems are chosen so that they cover different levels of difficulty. The performance of U-NSGA-III is compared to a generational real-parameter genetic algorithm (EliteRGA) which was used to solve various problems in the past [32], as well as CMA-ES [97] which is a state of the art algorithm for single objective optimization. We employ an elite-preserving operator between parent and offspring populations in the EliteRGA algorithm to make it equivalent to the degenerated form of U-NSGA-III for single-objective problems (that is, the $(\mu/2 + \mu)$ -ES form). Problem definitions are given in Equations 3.1, 3.2, 3.3, 3.4, 3.5 and 3.6.

$$f_{elp}(\mathbf{x}) = \sum_{i=1}^n ix_i^2, \quad (3.1)$$

$$-10 \leq x_i \leq 10, i = 1, \dots, n$$

$$f_{ros}(\mathbf{x}) = \sum_{i=1}^{n-1} [100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2], \quad (3.2)$$

$$-10 \leq x_i \leq 10, i = 1, \dots, n$$

$$f_{zak}(\mathbf{x}) = \sum_{i=1}^n x_i^2 + \left(\frac{1}{2} \sum_{i=1}^n ix_i \right)^2 + \left(\frac{1}{2} \sum_{i=1}^n ix_i \right)^4, \quad (3.3)$$

$$-1 \leq x_i \leq 1, i = 1, \dots, n$$

$$f_{sch}(\mathbf{x}) = 418.9829n - \sum_{i=1}^n (x_i \sin \sqrt{|x_i|}), \quad (3.4)$$

$$-500 \leq x_i \leq 500, i = 1, \dots, n$$

$$f_{ack}(\mathbf{x}) = -20e^{\left[\frac{-1}{5} \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2} \right]} - e^{\left[\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i) \right]} + 20 + e, \quad (3.5)$$

$$-32.768 \leq x_i \leq 32.768, i = 1, \dots, n$$

$$f_{ras}(\mathbf{x}) = 10n + \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i)], \quad (3.6)$$

$$-5.12 \leq x_i \leq 5.12, i = 1, \dots, n$$

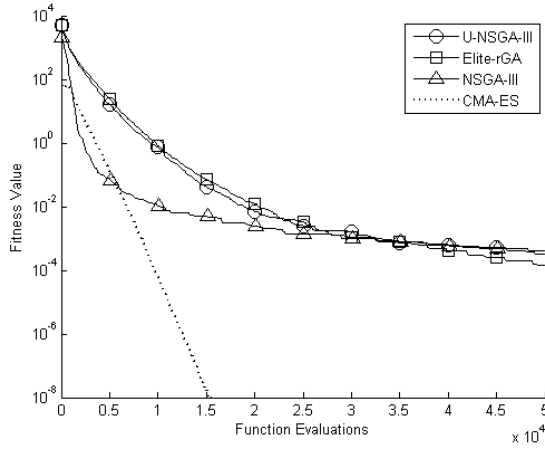
For each problem, $n = 20$ is used and 31 simulations with the same set of parameters but from different initial populations are conducted. Although the first three problems are simple, f_{sch} and f_{ras} are difficult multi-modal problems. We use $N = 48, 100, 100, 100$

and 300 for Ellipsoidal, Rosenbrock's, Zakharov's, Rastrigin's and Schwefel's problems, respectively.

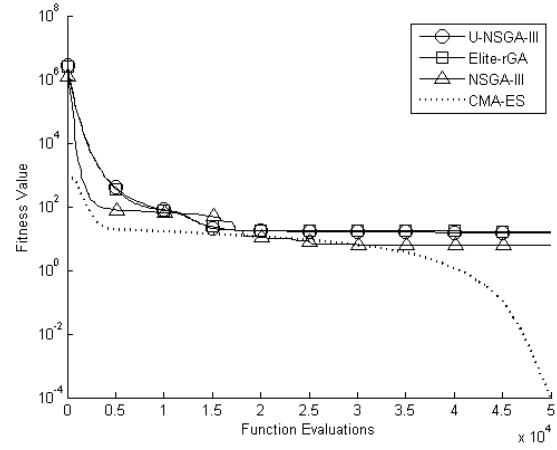
Figures 3.3a, 3.3b, S1, S2 and 3.3c show the median function value (y -axis) for Ellipsoidal, Rosenbrock's, Rastrigin's, Zakharov's and Schwefel's problems over 31 runs for different function evaluations (x -axis). EliteRGA, NSGA-III and U-NSGA-III perform similarly for unimodal and easy problems. However they are generally outperformed by CMA-ES in this category of problems. On the other hand, for the multi-modal problems U-NSGA-III and EliteRGA are the best, since both NSGA-III and CMA-ES are very prone to be trapped in local optima, as shown in Figures 3.3 Figure A.1 confirms our conclusion using Rastrigin's, a highly multi-modal function.

In order to confirm the superiority of U-NSGA-III over NSGA-III in multi-modal problems, we did the same comparison again over a range of polynomial mutation indices (η_m values), using two highly multi-modal test problems (Ackley's and Rastrigin's). Again, it is clear that NSGA-III is prone to get stuck into local optima because of its prohibitive restriction on population size. Even in the cases where NSGA-III is able to converge to the global optimum, we found it to be very sensitive to the polynomial mutation index (η_m). Only, a small (close to zero) η_m may enable NSGA-III to escape local optima and converge to the global optimum. This observation is clearly visible in Figures A.3 and A.4 for Ackley's and Rastrigin's, respectively.

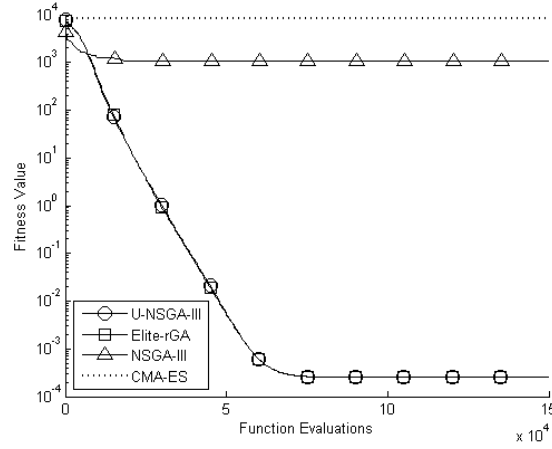
Table A.3 summarizes the best, median and worst fitness values achieved by each of the four algorithms on unconstrained test problems.



(A) Ellipsoidal function



(B) Rosenbrock's function



(C) Schwefel's function

FIGURE 3.3: Objective function value with respect to the number of function evaluations.

3.3.1.2 Constrained Problems

The superiority of U-NSGA-III and its equivalence to EliteRGA are more evident when it comes to constrained problems. The three algorithms under investigation have been tested against ten constrained test problems from the G-family test suite [98]. Best, median and worst achieved fitness are presented in Table A.4. Obviously, according to Figures A.5 and A.6, NSGA-III in most cases is not able to converge to the global optimum as expected. Again, U-NSGA-III and EliteRGA perform almost identically.

We conducted a Wilcoxon significance test over the medians of the final result of our 31 runs in each experiment. Tables A.1 and A.2 show the p – values of U-NSGA-III vs. NSGA-III, U-NSGA-III vs. EliteRGA and U-NSGA-III vs CMA-ES (the last comparison is conducted only for unconstrained test problems, because the original CMA-ES and its available implementation support only boxing constraints). Obviously, there is a statistically significant difference between U-NSGA-III and NSGA-III especially in multimodal problems (largest $p = 9.2668 \times 10^{-5}$). On the other hand, when comparing U-NSGA-III and EliteRGA these large p – values reported in the majority of our problems mean that we have to accept the null hypothesis. The null hypothesis simply means that the two distributions – from which the two sets of results are taken – have the same median. In other words, the two algorithms perform similarly without any statistically significant difference between them. It is also evident, that there is a statistically significant difference between U-NSGA-III and CMA-ES (largest $p = 5.7257 \times 10^{-8}$). However, according to the corresponding figures, this difference does not mean that one of them is better than the other in all problems. Actually, U-NSGA-III is found to be much better than CMA-ES in multimodal problems (see Rastrigin’s and Schwefel’s), while CMA-ES significantly outperforms U-NSGA-III in simpler problems (see Ellipsoidal, Rosenbrock’s and Zakharov’s). Similar conclusions can be reached for constrained test problems.

3.3.2 Bi-objective Problems

As mentioned before, the performance of NSGA-III was not tested on bi-objective optimization problems in the original study [4]. In this section, NSGA-II NSGA-III and

U-NSGA-III are compared over an extensive set of unconstrained and constrained bi-objective problems. Two types of experiments are shown here. These two experiments are supposed to help us draw conclusions about the niching mechanisms used in the three algorithms, and the effect of the niching-based selection operator of U-NSGA-III. We used ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6 as our unconstrained testbeds while, OSY, TNK, BNH and SRN are used as constrained test problems. Two additional real-world engineering problems – welded beam and pressure vessel design [49] – are also included. For all runs, we use SBX $p_c = 0.9$, $\eta_c = 30$ and polynomial mutation $p_m = 1/n$ and $\eta_m = 20$. In all ZDT problems, we use 30 variables and 31 different simulation runs are performed (except for ZDT6, where we use 10 variables, as used in the original study [99]).

3.3.2.1 N Equal to H

The first experiment compares the performance of the three algorithms when population size (N) is equal to the number of reference directions (H). By comparing the results of NSGA-II and NSGA-III only (ignoring U-NSGA-III for the time being), we can draw conclusions about the effectiveness of both NSGA-II and NSGA-III on bi-objective problems. Also, we can draw conclusions about the effect of the new niching-based selection operator introduced in U-NSGA-III by comparing the results of NSGA-III and U-NSGA-III only (ignoring NSGA-II).

Figures 3.4 and A.7 show the best, median and worst hypervolume (HV) achieved by each of the three algorithms on ZDT1, ZDT2, ZDT4 and ZDT6 (the figure of ZDT6 is removed for the sake of brevity). HV is calculated using an HV reference point 1% larger in every component than the corresponding *nadir* point.

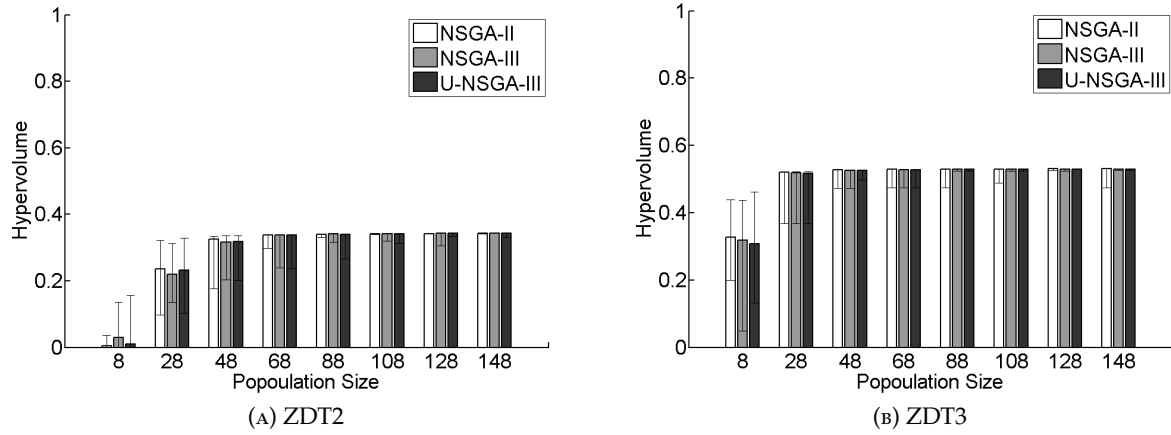


FIGURE 3.4: Performance of NSGA-II, NSGA-III, and U-NSGA-III with $N = H$ on unconstrained bi-objective test problems.

Although the results are comparable in most cases, NSGA-II's performance deteriorates significantly in ZDT4 compared to NSGA-III and U-NSGA-III especially at smaller population sizes. Table A.5 shows that the three algorithms are close to each other. However, from Table A.8, we can see that in ZDT3 and ZDT6, there is a statistically significant difference in favor of NSGA-II, while in ZDT4, U-NSGA-III significantly outperforms NSGA-II.

The same observation can be noticed for constrained test problems in Figure A.8.

The more difficult the problem is (BNH and SRN), the bigger the difference in favor of NSGA-III and U-NSGA-III. It is also clear that NSGA-II is the most negatively affected by very small population sizes (namely, $N = 8$), while both NSGA-III and U-NSGA-III are more robust with respect to small population sizes (see Table A.6).

The same observation can be made in Figures A.9 and A.10 showing the results of welded beam and pressure vessel problems, respectively. The ideal and reference points used to compute HV values for constrained problems are shown in Table 3.3.

TABLE 3.1: Threshold HV values used in bi-objective ZDT problems for $N \geq H$ simulations.

Problem	Fixed HV	Problem	Fixed HV
ZDT1	0.640	ZDT4	0.530
ZDT2	0.316	ZDT6	0.390
ZDT3	0.512		

The statistical significance test results (Table A.9) shows that there is a statistically significant difference in favor of U-NSGA-III over NSGA-II in BNH, SRN and Pressure Vessel. The opposite however is never valid.

3.3.2.2 N Greater Than H

In the second set of experiments for handling bi-objective problems, we evaluate the usefulness of using $N \geq H$. In this experiment, two straight lines, one representing the performance of NSGA-II with $N = 16$ and another representing the performance of NSGA-III with $N = H = 16$ are shown for convenience. The jagged line represents the performance of U-NSGA-III with $N \geq H$ and $H = 16$.

The criterion of comparison used here is the number of function evaluations required to reach a pre-defined threshold hyper-volume (HV) value. Table 3.1 presents the HV values used in our bi-objective simulations.

For U-NSGA-III, we have used different population sizes $N \geq H$. Average numbers of function evaluations over 31 runs – needed to achieve the prespecified HV – are plotted in Figures 3.5 and A.11 for unconstrained ZDT test problems. Constrained problems results are shown in Figure A.12.

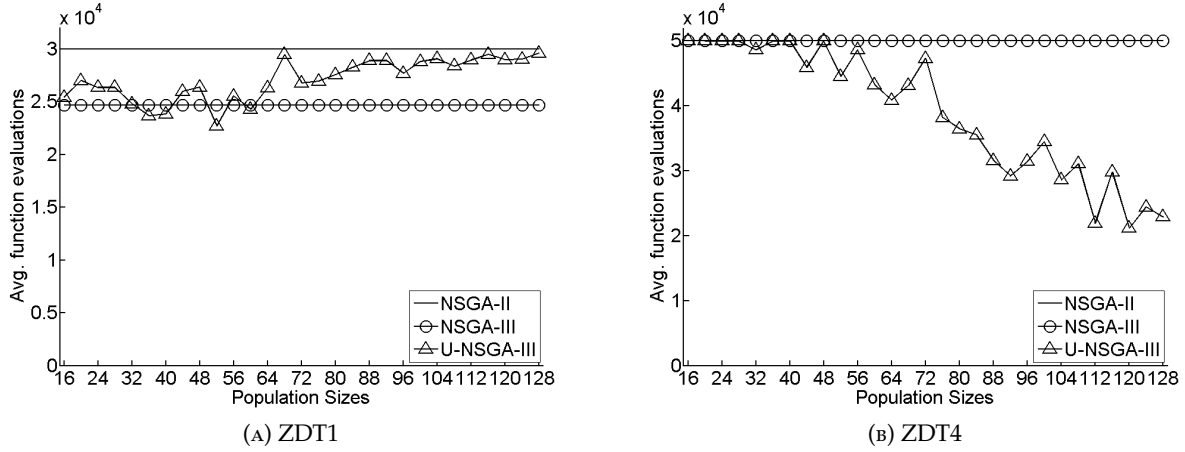


FIGURE 3.5: Performance of NSGA-II, NSGA-III, and U-NSGA-III with $N \geq H$ on unconstrained bi-objective ZDT test problems.

For runs having a population size larger than number of reference directions, out of all the individuals attached to a reference direction, only one contributes to the HV calculation (the closest). that is, for all U-NSGA-III simulations, only 16 individuals one closest to each specified reference direction are used to calculate the final HV value, no matter what population size is used. We did so to retain our ability to compare HV values for all simulations with different population sizes. For ZDT1, ZDT3 and TNK, the use of a larger population size is not found to be beneficial, whereas for ZDT2, ZDT4, BNH and SRN (more difficult problems), a larger population brings in the necessary diversity needed to solve the respective problem adequately. It is clear that in all problems, there exists certain population sizes, in general, higher than H that make U-NSGA-III to perform better than NSGA-III and NSGA-II. For relatively difficult problems, the difference is quite obvious. In most cases, however, the performance of NSGA-III is better than NSGA-II, due to the use of an external guidance for diversity through a uniformly distributed set of reference directions. These results are interesting and demonstrate the usefulness of a larger population size than the number of reference directions for the proposed

U-NSGA-III algorithm.

3.3.3 Three-objective Problems

To enable U-NSGA-III to work well on single and bi-objectives, there should not be any performance degradation to three and many-objective problems. In this section, we present results on three-objective unconstrained DTLZ1, DTLZ2, scaled DTLZ1 and scaled DTLZ2 problems. We also include the two constrained test problems C3-DTLZ1 and C3-DTLZ4 proposed in [7]. In this section, U-NSGA-III is compared to both NSGA-II and NSGA-III.

The performance metric used for these problems is also the hypervolume metric, but due to the increased computational efforts in extending the hypervolume computation to many-objective problems, we use the fast technique proposed elsewhere [100]. It is understood that DTLZ problems have mathematically defined description of their efficient fronts, thereby making it possible for us to compute the theoretical hypervolume if infinite points are put on the true efficient front. For DTLZ1 problem, the efficient front is a M -dimensional linear hyperplane making equal angle with all objective axis and intersecting each axis at 0.5. Thus, the efficient front is a M -simplex in M -dimensional space and the volume under the front is given as follows [101]: $V_1(M) = (1/M!)(0.5^M)$. Therefore, the theoretically maximum hypervolume for a reference points at $\mathbf{z} = (1 + \epsilon)(0.5, 0.5, \dots, 0.5)^T$ is

$$HV_T = (0.5(1 + \epsilon))^M - \frac{1}{M!}0.5^M. \quad (3.7)$$

The normalized hypervolume for a set of non-dominated individuals P is then defined from the calculated hypervolume $HV(P)$, as follows:

$$HV_{norm} = \frac{HV(P)}{HV_T}. \quad (3.8)$$

For DTLZ2 problem, the efficient front is a part of the M -dimensional hypersphere of radius one and the volume under the efficient front is given as follows [102]:

$$V_2(M) = \begin{cases} \frac{\pi^{M/2}}{2^M(M/2)!}, & \text{if } M \text{ is even,} \\ \frac{(\pi/2)^{(M-1)/2}}{M(M-2)(M-4)\dots 1}, & \text{if } M \text{ is odd.} \end{cases}$$

For a reference point $\mathbf{z} = (1 + \epsilon)(1, 1, \dots, 1)^T$, the theoretical hypervolume is given as follows:

$$HV_T = (1 + \epsilon)^M - V_2(M), \quad (3.9)$$

and the normalized hypervolume can be computed by using Equation 3.8. Table 3.2 presents HV_T values for a few M values for both DTLZ1 and DTLZ2 for $\epsilon = 0.01$.

TABLE 3.2: The theoretical hypervolume HV_T for DTLZ1 and DTLZ2 for 3, 5, 8, 10 and 15 objectives ($\epsilon = 0.01$)

Problem	Objective dimension, M				
	3	5	8	10	15
DTLZ1	0.107954	0.032584	0.004230	0.001079	0.000035
DTLZ2	0.506702	0.886517	1.067002	1.102132	1.160957

For solving three-objective problems, we have used $N = 92$ for all three algorithms and $H = 91$ for U-NSGA-III and NSGA-III. Each figure represents the best of 11 distinct runs. Figure A.15 shows how the three algorithms perform on DTLZ1 and DTLZ2.

TABLE 3.3: Ideal and reference points used for constrained bi-objective problems.

Problem	Ideal Point	Reference Point
OSY	$(-275, 5)$	$(-40.4, 77.77)$
TNK	$(0.029, 0.029)$	$(1.0605, 1.0605)$
BNH	$(0, 3.667)$	$(138.407, 50.5)$
SRN	$(5, -215)$	$(227.25, 0)$
Welded	$(0, 0)$	$(40.4, 0.00505)$
Pressure	$(42, -62761000)$	$(330270, -8080)$

Results of the scaled versions of the two problems are shown in Figures 3.6a, 3.6b, 3.6c, 3.6d, 3.6e and 3.6f, respectively.

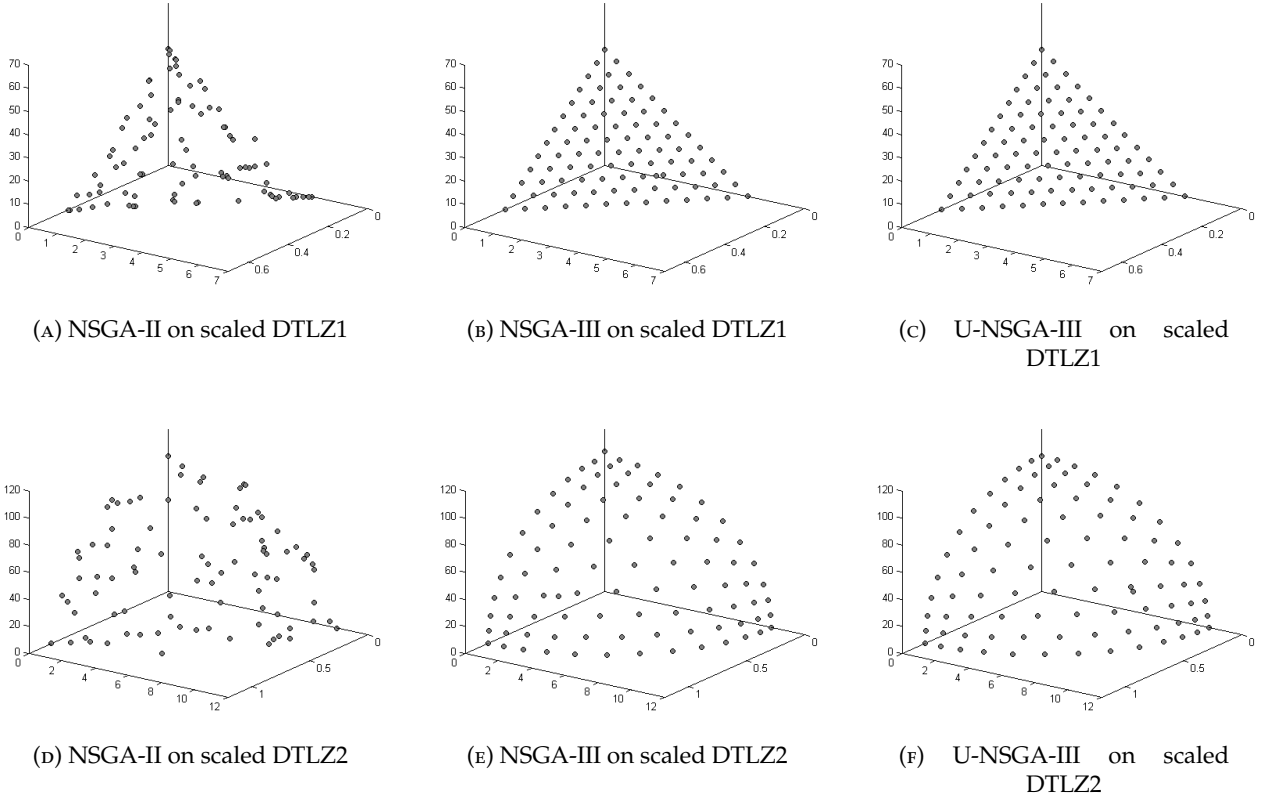


FIGURE 3.6: Performance of NSGA-II, NSGA-III, and U-NSGA-III on scaled unconstrained three-objective DTLZ problems.

Finally, Figure A.16 presents the final population of the three algorithms for problems C3-DTLZ1 and C3-DTLZ4.

TABLE 3.4: Best, medium and worst HV(norm) on multi/many-objective unconstrained DTLZ problems.

Problem	Obj.	G	P	NSGA-II			NSGA-III			U-NSGA-III		
				Best	Median	Worst	Best	Median	Worst	Best	Median	Worst
DTLZ1	3	400	92	.9198	.9136	.8597	.9487	.9465	.9388	.9462	.9464	.934
	5	600	212	—	—	—	.9767	.9762	.9757	.9766	.9760	.9751
	8	750	156	—	—	—	.9953	.9953	.9953	.9953	.9953	.9953
	10	1000	276	—	—	—	.9972	.9972	.9972	.9972	.9972	.9972
DTLZ2	3	250	92	.8976	.8830	.8711	.9828	.9819	.9812	.9838	.9824	.9808
	5	350	212	.3763	.3143	.2072	.8407	.8396	.8371	.8404	.8398	.8382
	8	500	156	—	—	—	.8532	.8492	.8452	.8525	.8497	.847
	10	750	276	—	—	—	.8769	.8760	.8743	.8769	.8751	.8743
S-DTLZ1	3	400	92	.9204	.9111	.8993	.9488	.9482	.9456	.9485	.9472	.9445
	5	600	212	—	—	—	.9767	.9762	.9675	.9768	.9764	.9565
	8	750	156	—	—	—	.9946	.9941	.9931	.9943	.9941	.9920
	10	1000	276	—	—	—	.9991	.9991	.9981	.9991	.9991	.9981
S-DTLZ2	3	250	92	.8034	.7914	.7739	.8756	.8741	.8715	.8749	.8739	.8705
	5	350	212	.3761	.2895	.2376	.8393	.8349	.8314	.8384	.8353	.8285
	8	500	156	—	—	—	.8510	.8485	.8463	.8512	.8490	.8459
	10	750	276	—	—	—	.9218	.9173	.9066	.9228	.9200	.8935

It can be seen from the figures that while NSGA-II fails to maintain adequate distribution of individuals, both NSGA-III and U-NSGA-III successfully achieve a uniformly distributed set of individuals covering the entire efficient front in each case. Only minor differences can be seen between the plots of NSGA-III and U-NSGA-III. Best, median and worst HV values in Tables 3.4 and A.7 show the equivalence in performance between NSGA-III and U-NSGA-III, as anticipated. For scaled problems, we first unscale the objective values using the scaling factor used in the optimization process and then compute HV_{norm} metric value.

3.3.4 Many-objective Problems

Finally, we consider five, eight, and 10-objective versions of the same 6 problems used in the previous subsection. We compare our proposed U-NSGA-III with NSGA-III (with N equal to H) in using hyper-volume values, which are computed using the sampling based strategy proposed elsewhere [100] due to the computational complexities involved in HV

calculations in higher dimensions. The differences between U-NSGA-III and NSGA-III were analyzed to be negligible for many-objective optimization problems from an algorithmic point of view. Here, we investigate how both these methods perform empirically on a series of test problems. In these problems, we use the same N and H values used in the original NSGA-III study.

Tables 3.4 and A.7 clearly show that NSGA-II fails in terms of both convergence and maintaining diversity, at the level of five or more objectives, to the extent that none of the individuals of the final population passed the reference point used to calculate HV. On the other hand, NSGA-III and U-NSGA-III produce very similar HV values in all problems. Almost-identical Parallel Coordinate Plots (PCP) can be observed for the 10-objective versions of DTLZ problems in Figures A.13, 3.7 and A.14. A PCP is a set of vertical bars, each represents one objective. Each zigzag horizontal line represents one individual. The point at which a zigzag line cuts a vertical bar represents the corresponding objective value of the corresponding individual. A PCP plot gives a rough picture of the diversity of set of solutions. The more intersections and the more coverage of the whole plot, the better.

Finally, Table A.10 shows that from a statistical point of view, U-NSGA-III and NSGA-III are equivalent on almost all multi/many-objective problems up to ten objectives. which means that the new niching based operator did not affect NSGA-III ability to tackle higher dimensional problems.

All these results demonstrate that the introduction of the niched tournament selection in the original NSGA-III algorithm and the flexibility of using a different population size

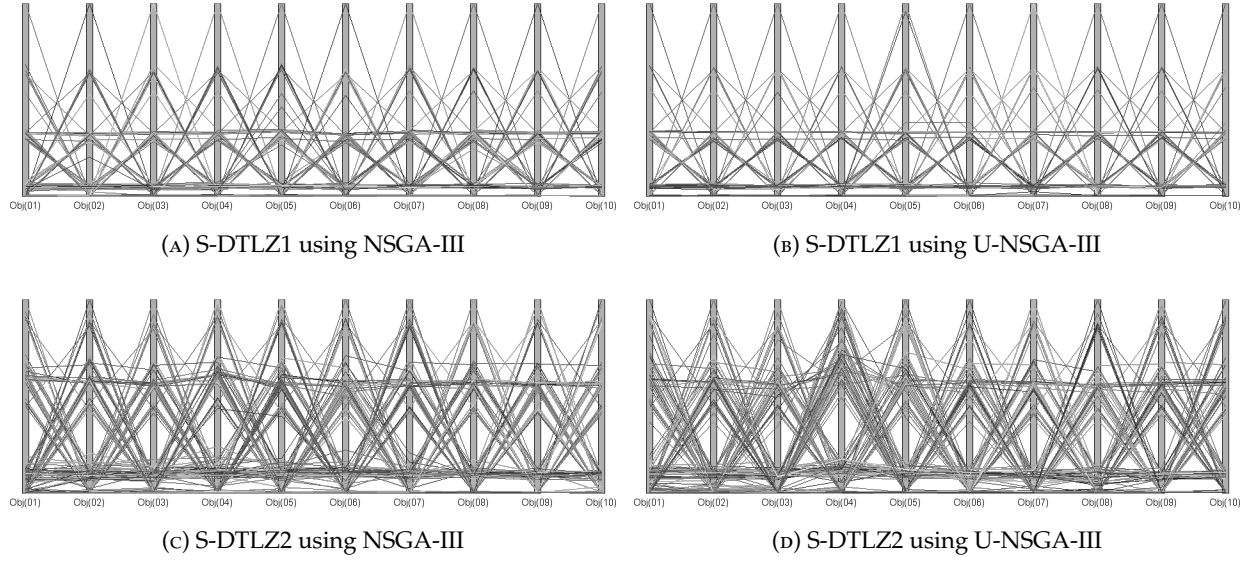


FIGURE 3.7: Performance of NSGA-III and U-NSGA-III on scaled unconstrained many-objective DTLZ problems.

from the number of reference directions do not change its performance in U-NSGA-III on many-objective optimization problems.

3.4 Summary

In this chapter, we developed a unified evolutionary optimization algorithm U-NSGA-III which is a modification of a recently proposed evolutionary many-objective optimization method. U-NSGA-III has been carefully designed to solve single, multi-, and many-objective optimization problems. Simulation results on a number of single, two, three, five, eight and ten-objective constrained and unconstrained problems have demonstrated the efficacy of the proposed unified approach. In each category having multiple problem instances, it has been found that the proposed U-NSGA-III performs in a similar manner and sometimes better than a respective specific EA – an elite-preserving RGA for single-objective problems, NSGA-II for bi-objective problems, and NSGA-III for three

and many-objective problems. The ability of one optimization algorithm to solve different types of problems equally efficiently and sometimes better, with the added flexibility brought in through a population size control remains a hallmark achievement of this study. In addition, several useful insights have been elaborated on, both algorithmically and empirically about the relative effectiveness of all the algorithms included in this chapter.

This type of unification has not been attempted before, except in omni-optimizer approach [103] which is not scalable for many-objective problems. In this regard, this study makes a key contribution in suggesting one single optimization algorithm that is able to degenerate into efficient single, multi- and many-objective optimization methods. dictated simply by the number of objectives in a given problem. Due to these reasons, the study is important from the efficient optimization software development point of view and its applicability to practical problems having separate one and many-objective versions. Such unification approaches also provide researchers the key insight about operator interactions needed to constitute scalable algorithms.

The unified optimization algorithm proposed here elevates the act of optimization as a computing-friendly approach. Computing algorithms are usually developed for handling generic inputs having large-dimensional attributes or parameters. However, the algorithm is also expected to work on a specific lower-dimensional or trivial input as a degenerate case of the generic case. For example, the Gauss-elimination computing algorithm was developed to solve a multi-variable linear system of equations $\mathbf{Ax} = \mathbf{b}$, but if the same algorithm is used to solve a single-variable linear equation, $ax = b$ (a degenerate case), the algorithm should find the solution $x = b/a$ without any hitch. In the same way,

depending on the number of objectives, U-NSGA-III attempts to find multiple Pareto-optimal solutions if the objectives are greater than one, but when the number of objectives is only one, U-NSGA-III finds a single optimal solution as a degenerate case.

It is obvious from all the results presented earlier that U-NSGA-III performs as expected in both convex and non-convex continuous real-variable single, bi-, multi- and many-objective optimization problems. And although some problems have discontinuous Pareto fronts (like ZDT3) the whole search space of the problem remains continuous. A proper extension to this study is to test the performance of U-NSGA-III on discrete or generally discontinuous problems. Also, many EMO algorithms – including NSGA-III and U-NSGA-III – tend to ignore combinatorial optimization despite the fact that it is one of the most important categories of optimization.

Because of the important role selection plays in NSGA-III and U-NSGA-III, in the next chapter, we further explore how selection affects this type of algorithms.

Chapter 4

Effect of Selection Operator on NSGA-III in Single, Multi, and Many-Objective Optimization

Chapter 3 shows clearly the effect selection has on the performance of NSGA-III, and how it is the key behind the superiority of U-NSGA-III in lower dimensions. This urged us to further explore how selection changes the behavior of the algorithm across different types of problems. In this chapter, we examine the possibility of ignoring the single fold restriction ($N = H$ when applying NSGA-III to different types of problems. Performance of the unrestricted (multi-fold) NSGA-III is compared to the performance of U-NSGA-III over a wide range of constrained and unconstrained problems. We propose two main hypotheses in this study and try logically and empirically to validate their correctness.

As shown in Chapter 3, the larger the difference $(N - H)$, the more selection we need. Unlike NSGA-III, U-NSGA-III is free from the $N = H$ restriction. The frequency and effect of selection depends on N and H relative values. For example, in single-objective scenarios, where $H = 1$ and $N \gg H$, selection will be performed at all times, because all individuals will belong to the same unique reference direction. On the other extreme, in many-objective scenarios, where $N = H$ was restricted to make the algorithm computationally efficient, each solution is expected to be attached to a different direction, meaning that selection is practically absent. In the U-NSGA-III study, we were able to further show the efficiency of U-NSGA-III in single, multi and many-objective problems and for both constrained and unconstrained situations due to its adaptive selection operator. We also compared the proposed unrestricted U-NSGA-III with the already existing restricted NSGA-III (where $N = H$). None of the simulations tested the performance of NSGA-III if the single-fold restriction is ignored.

In Section 4.1 we propose our two hypotheses and justify each through arguments. Results of our extensive simulations confirming our logical arguments are presented in Section 4.2. Finally, Section 4.3 concludes this study.

4.1 Hypotheses

From the previous section, we can see that the two main features characterizing NSGA-III are the absence of selection and the usage of reference directions as an external guidance mechanism to preserve diversity among solutions. In the following two subsections, our two main hypotheses are stated and discussed in detail in highlight of the absence of selection in NSGA-III and its effect when N is significantly larger than H .

4.1.1 NSGA-III Convergence

As long as the population size is equal to the number of reference directions, the absence of selection is not expected to affect the results, because each reference direction is expected to attract only one point as they go further in generations. However, the situation becomes different if N is greater than H . In such a case, some population members will be guided by the reference directions while the others will keep floating randomly in the search space due to lack of any selection pressure guiding anywhere in the search space. No selection to decide between individuals and not enough reference directions to guide the whole population, this effect should be more evident as the gap between N and H becomes bigger. We call this behavior "excessive randomness", and it leads us to our first hypothesis.

Hypothesis 1. In NSGA-III, the use of a population size larger than the number of reference directions slows down convergence.

A direct consequence of hypothesis 1 is that NSGA-III should be converging slower than any equivalent evolutionary algorithm involving any form of selection that honors better solutions. This very same fact can be stated differently by saying that NSGA-III tends to be less greedy than other selection-based evolutionary algorithms.

4.1.2 NSGA-III and Local Optima

Although NSGA-III is expected to be slower than its selection-based counterparts, this is not always a disadvantage. Actually, this might be beneficial in some cases. Because of the inherent diversity of solutions maintained through "excessive randomness" in NSGA-III

(discussed above), the algorithm is expected to have higher ability to escape local optima. It is also expected to be less-dependent on mutation operators. Again, this behavior is justified, owing to the less-greedy nature of the algorithm. Hence, we can state our second hypothesis.

Hypothesis 2. If $N > H$, NSGA-III will have a higher ability to escape local optima than its selection-based counterparts, and consequently it becomes less dependent on mutation operators.

In Section 4.2, our simulation results are presented and discussed in the highlight of these two hypothesis. Besides supporting our hypotheses, we also paint a clear picture of the performance of multi-fold NSGA-III with single, multi, and many-objective optimization problems.

4.2 Results

In this section we conduct two experiments. In both experiments, we compare NSGA-III with U-NSGA-III. As mentioned earlier, U-NSGA-III is a variant of NSGA-III with the additional niching-based selection operator. We previously showed that U-NSGA-III in single-objective scenarios becomes equivalent to a standard $(\mu + \lambda)$ evolutionary strategy (ES) [95]. The parameters used in our single-objective simulations are shown in Table 4.1. For each problem, $n = 20$ (number of decision variables) is used. The results shown in all the plots of this study are the medians of 11 simulations with the same set of parameters. Regarding higher dimensions, U-NSGA-III and NSGA-III have been compared in the first study, abiding by the condition $N = H$ in both algorithms. Using such a setting,

U-NSGA-III degenerates to NSGA-III after its additional selection operator loses its effect. Here in all our comparisons we use the same number of population folds for both algorithms. This will enable us to investigate the effect of selection in higher dimensions as well.

TABLE 4.1: Parameter listing.

Parameter	Value
SBX distribution index (η_c)	0
Polynomial mutation distribution index(η_m)	20
Crossover Probability	0.75
Mutation Probability	0.02

4.2.1 Single-Objective Problems

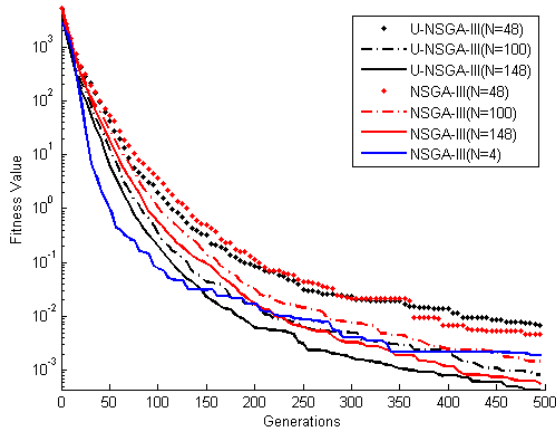
The first experiment is to compare both NSGA-III and U-NSGA-III using multiple population folds. In single objective problems, only one reference direction exists, which is the extreme of the $N \gg H$ case. Our simulations are performed on a group of unconstrained as well as constrained problems spanning a wide range of difficulties. The unconstrained test problems are Ellipsoidal, Rosenbrock's, Schwefel's, Ackley's and Rastrigin's listed in Equations 3.1, 3.2, 3.4, 3.5 and 3.6, while the constrained problems are G01, G02, G04, G06, G07, G08, G09, G10, G18 and G24 form the standard G-test suite [98]. Each figure compares both methods for three different population sizes; 100, 300 and 500. The only exception is the relatively easy Ellipsoidal problem, for which we used $N = 48, 100$ and 148. Log scale is used whenever necessary to show performance differences in late generations. It is clear from Figures 4.1, 4.2 and 4.3 that NSGA-III converges *slower* than U-NSGA-III in both constrained and unconstrained problems across all three population sizes, thereby confirming the correctness of our first hypothesis 1. In most cases, NSGA-III

is able to catch up with U-NSGA-III if enough time (or generations) is given. In some cases, especially in constrained problems, like G01, G06, G07 and G10, NSGA-III needs almost twice the number of generations needed by U-NSGA-III to achieve the same level of convergence. Of course, U-NSGA-III can only save algorithmic time and has no effect on the time spent on function evaluations.

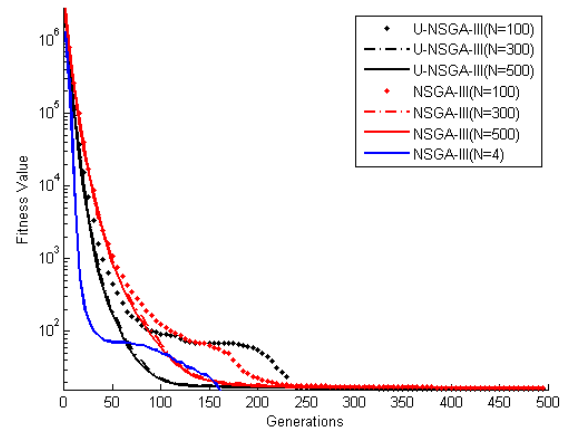
We have also included the median GD per generation of 11 NSGA-III runs where $N = H$ (blue line), in order to feel the usefulness multiple folds compared to a single fold. For the blue line, the x-axis does not represent the number of generations anymore. However, to maintain a fair comparison, at any given point on the x-axis, the blue line and the dotted black and red lines have the same number of solution evaluations. The figures show that NSGA-III where $N = H$ tends to converge very quickly at the beginning then it gets stuck after a while without reaching the global optimum. This is expected because of the too small population size ($N=4$) which makes the algorithm prone to be trapped in local optima even if given the same number of solution evaluations as the multiple folds version of the algorithm.

4.2.2 Multi and Many-Objective Problems

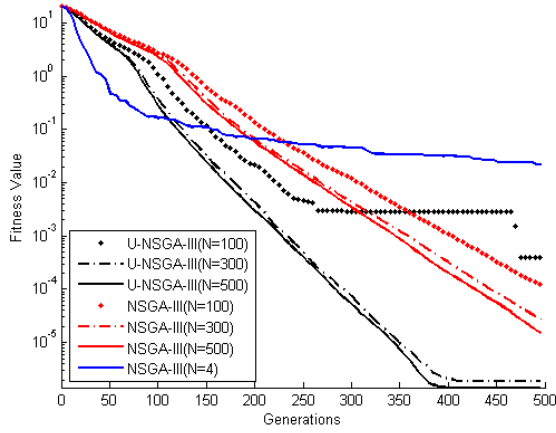
In higher dimensions however, diversity becomes as important as convergence. Since Hypothesis 1 relates only to convergence speed, we needed a way of testing only the convergence ability of each algorithm. Having this in mind, we use Generational Distance (GD) as our metric. Although, in some cases, GD does not reach Zero (full convergence), it can still show the relative difference in performance. Another important aspect is problem selection. Our test problems should primarily test the convergence ability of the



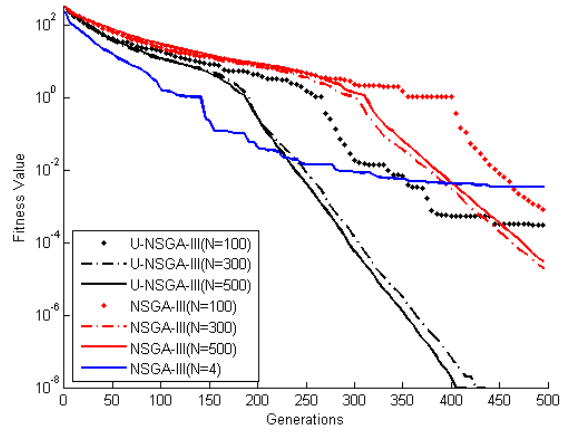
(A) Ellipsoidal



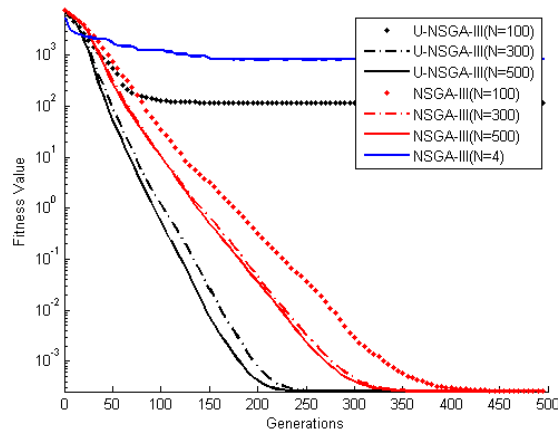
(B) Rosenbrock's



(C) Ackley's

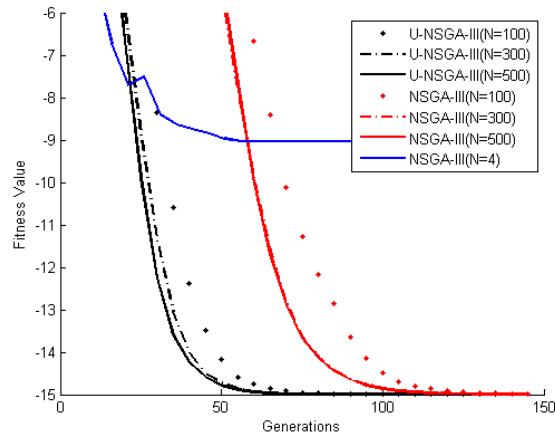


(D) Rastrigin's

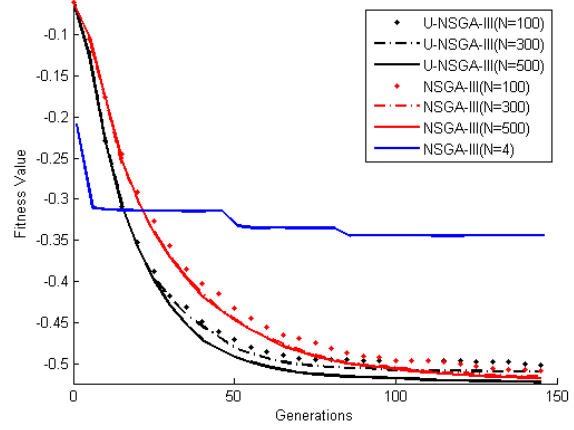


(E) Schwefel's

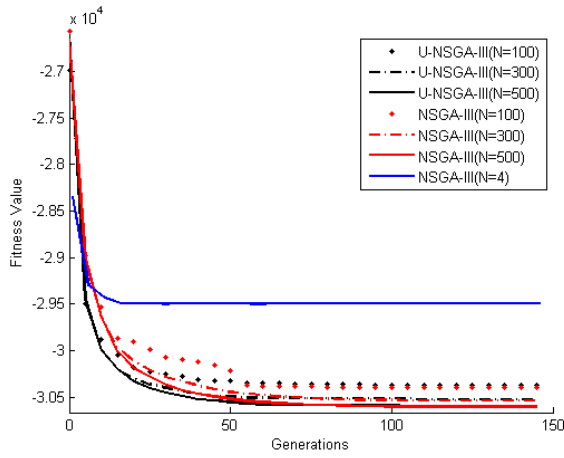
FIGURE 4.1: Single-objective Unconstrained Problems.



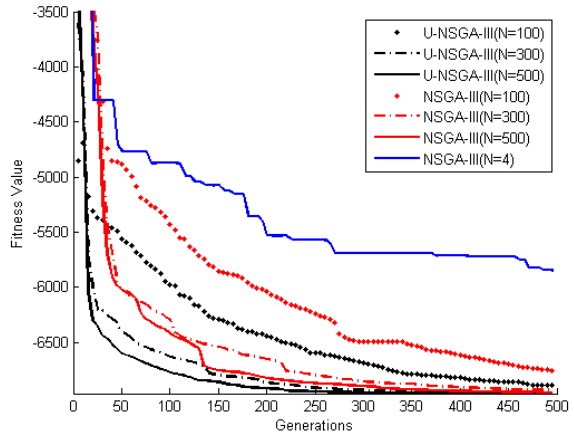
(A) G1



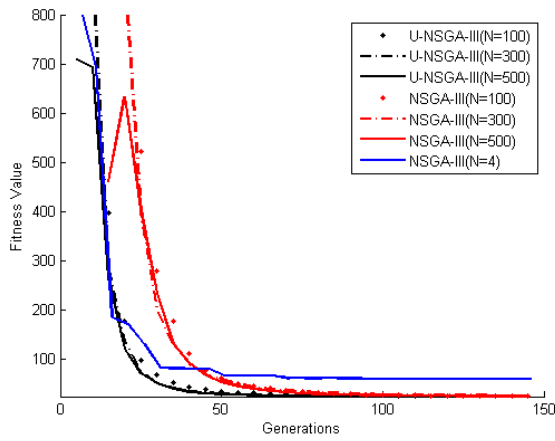
(B) G2



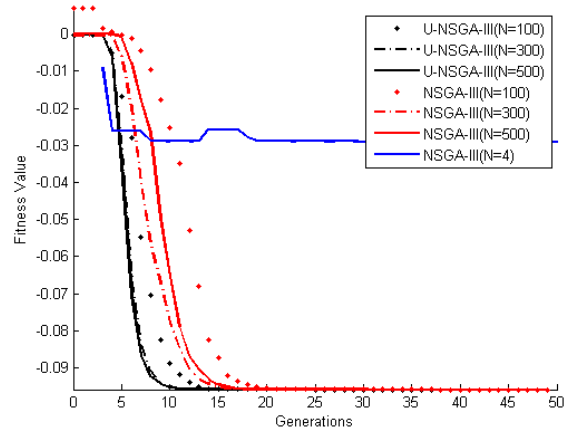
(C) G4



(D) G6



(E) G7



(F) G8

FIGURE 4.2: Single-objective Constrained Problems (G1, G2, G4, G6, G7 and G8).

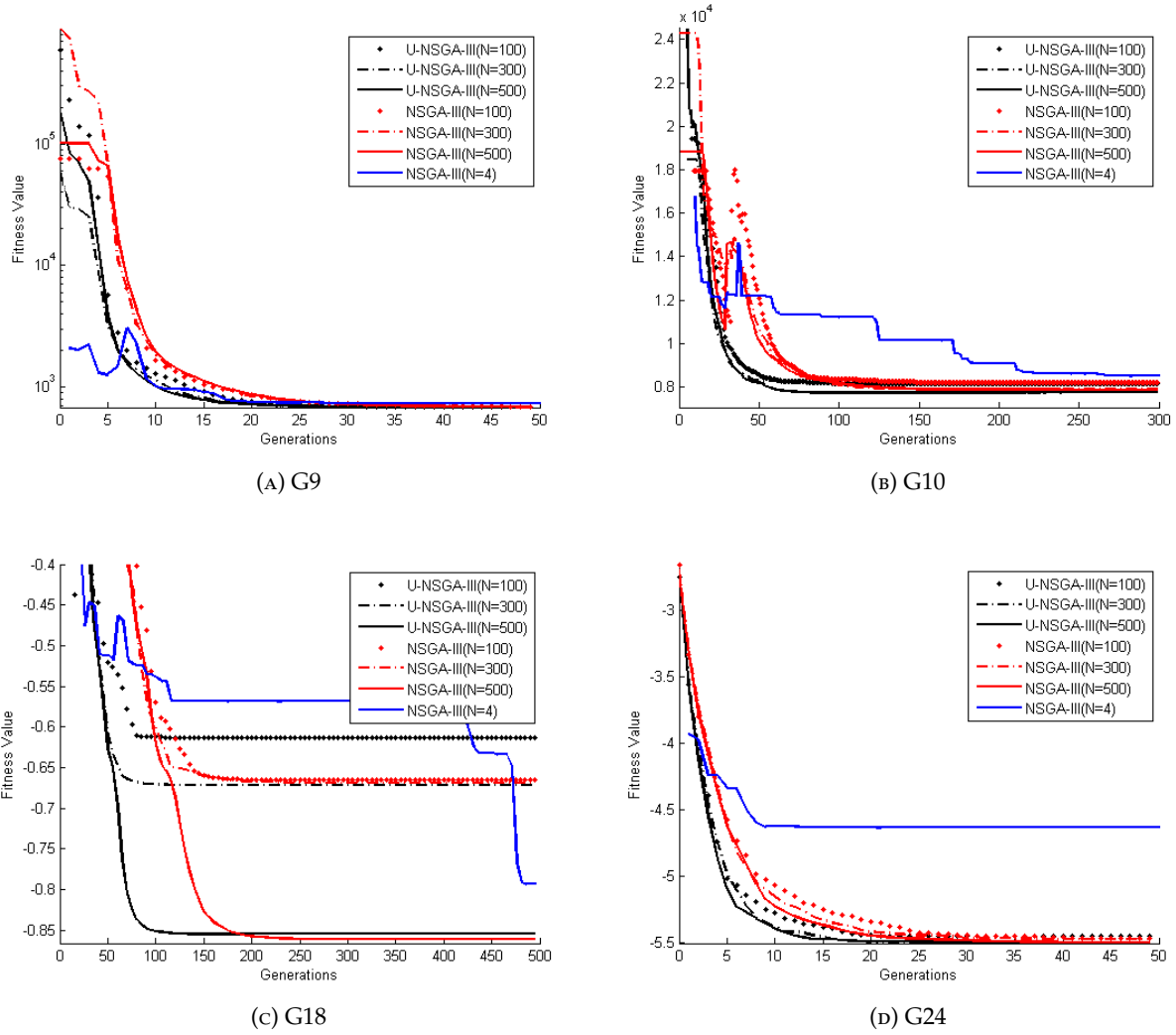


FIGURE 4.3: Single-objective Constrained Problems (G9, G10, G18 and G24).

algorithm. For this reason, we choose ZDT4 [99] as our bi-objective test bed. ZDT4 has $(21^9 - 1)$ local Pareto optimal fronts. Unless the convergence ability of the algorithm is good, it will get trapped in one of these local fronts. We also included two easy problems, ZDT1 and ZDT2 to show the effect of selection in simple and easy problems.

For three and five objectives, we choose DTLZ1 [104], which has $(11^5 - 1)$ local Pareto-optimal fronts, requiring good convergence ability as well.

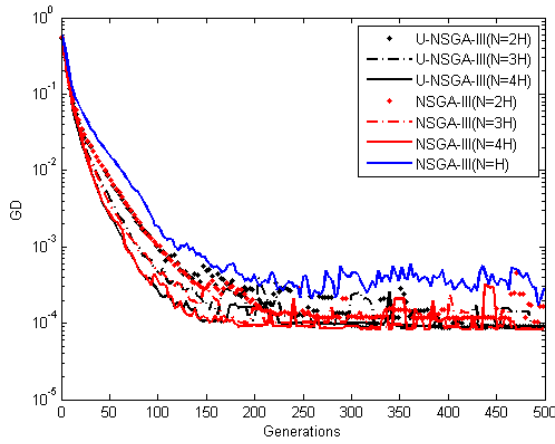
In simple and easy problems containing no local Pareto-optimal fronts, both the algorithms

exhibit almost identical convergence behaviors, as shown in Figures 4.4a and 4.4b. This is expected, because in order for the difference to show up between these algorithms, the problems should exhibit adequate difficulty preventing them from reaching the global Pareto-optimal front. This difficulty is introduced in ZDT4. Figure 4.4c shows the median GD metric of 11 runs for 500 generations using $N = 1.5H$, $N = 2H$ and $N = 2.5H$. It is clear from the figure that U-NSGA-III converges faster than NSGA-III. The difference between the two algorithms becomes more obvious from Figures 4.5a and 4.5b. Final fronts of the best, median and worst runs for the two simulations ($N = 1.5H$) and ($N = 2H$) are shown in these two figures, respectively. The difference is negligible in the best runs. It increases gradually becoming visible at median runs, and relatively large at worst runs. Using a smaller population size increases the difference in convergence between the two algorithms. Figures 4.6a and 4.6b show the behavior of the GD metric for DTLZ1 in three and five-objective problems, respectively. Although, U-NSGA-III still maintains a slight edge over NSGA-III in terms of convergence, the difference is small. This can be attributed to the fact that convergence becomes less important compared to diversity maintenance as the number of objectives increases. All these results clearly support the correctness of our first hypothesis 1 for bi-objective and multi-objective problems.

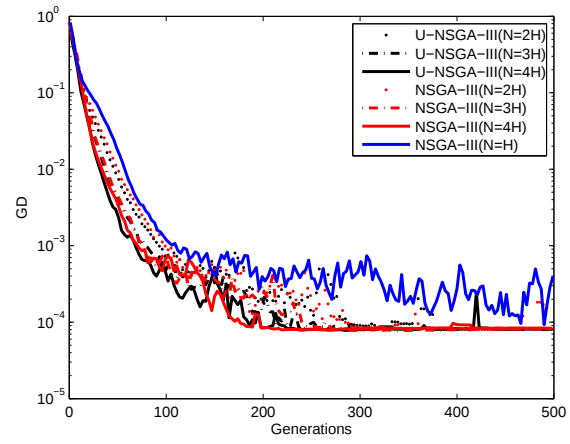
Again, in order to justify using multiple folds in multi-objective scenarios, median GD plots of 11 runs on ZDT1, ZDT2 and ZDT4 (single fold) are added to Figures 4.4a, 4.4b and 4.4c respectively. Obviously, using only single fold NSGA-III prevents the algorithm from converging to the true Pareto front.

In ZDT4, it is obvious that as fold size grows (1.5, 2, 2.5 etc.), NSGA-III's convergence ability approaches that of U-NSGA-III. This has nothing to do with hypothesis 1, it is

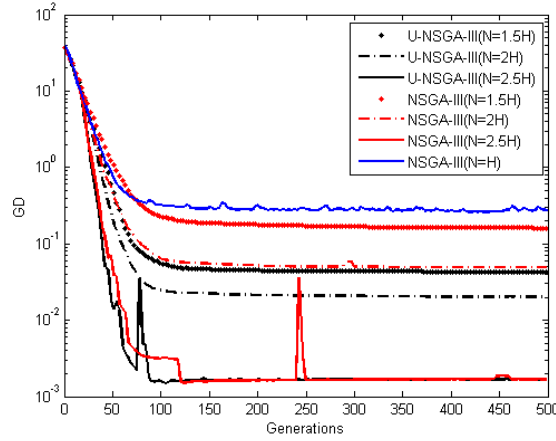
actually a clear consequence of hypothesis 2. Since ZDT-4 has so many local fronts, an algorithm with higher ability to escape these local optima is preferable. This is exactly the case here. Although, multi-fold NSGA-III suffers from being slow, its ability to avoid local optima increases as the number of folds increases. This compensates for its slowness and enables it to catch up with U-NSGA-III. More on hypothesis 2 in Section 4.2.3.



(A) ZDT-1 Median GD (H=25, N=2H, 3H, 4H)

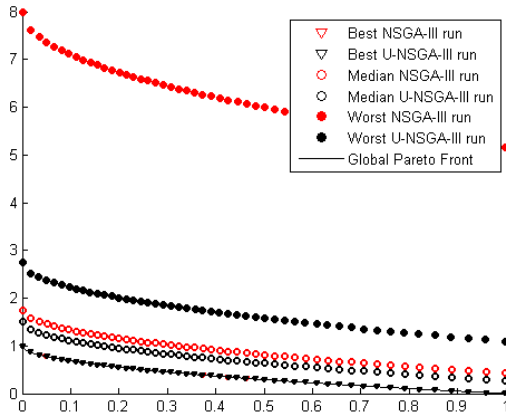


(B) ZDT-2 Median GD (H=25, N=2H, 3H, 4H)

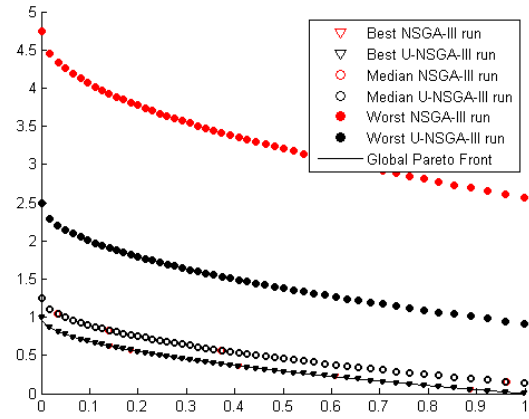


(C) ZDT-4 Median GD (H=50, N=1.5H, 2H, 2.5H)

FIGURE 4.4: Bi-objective Unconstrained Problems (GD)

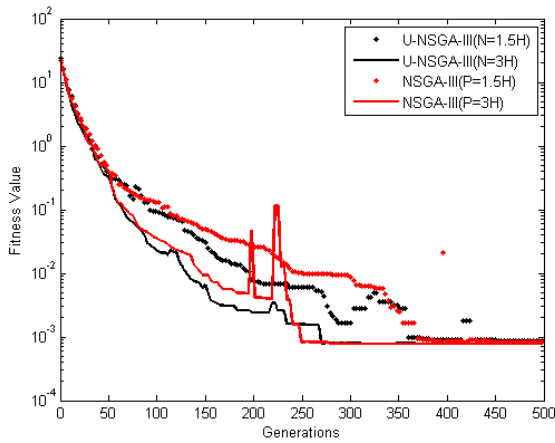


(A) ZDT-4 Final Generation ($H=50$, $N=1.5H$)

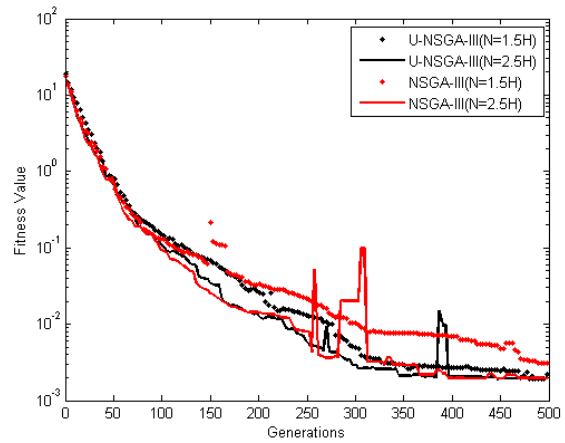


(B) ZDT-4 Final Generation ($H=50$, $N=2H$)

FIGURE 4.5: Bi-objective Unconstrained Problems (fronts)



(A) DTLZ-1 (3 Obj.) Median GD ($H=55$, $N=1.5H$, $3H$)



(B) DTLZ-1 (5 Obj.) Median GD ($H=73$, $N=1.5H$, $2.5H$)

FIGURE 4.6: Multi-/Many-objective Problems

4.2.3 Multi-modal Problems

Our second experiment tests the ability of NSGA-III to escape local optima and its dependence on the mutation operator compared to U-NSGA-III. Again, U-NSGA-III is used here as a candidate single-objective algorithm involving selection. For our purpose, we choose three highly multi-modal problems, namely, Ackley's, Rastrigin's and Schwefel's.

A relatively small population size ($N = 48$) is used to make both the algorithms more prone to be trapped in local optima. We set mutation probability to zero, thus completely removing mutation from both. Now the ability of both algorithms to escape local optima by themselves – without any help from a mutation operator – can be clearly observed. As shown in Figure 4.7, the ability of NSGA-III to escape local optima is evident compared to U-NSGA-III, which got easily trapped in local optima in the absence of mutation. This goes hand in hand with the less-greedy nature of NSGA-III discussed earlier, and confirms the correctness of Hypothesis 2.

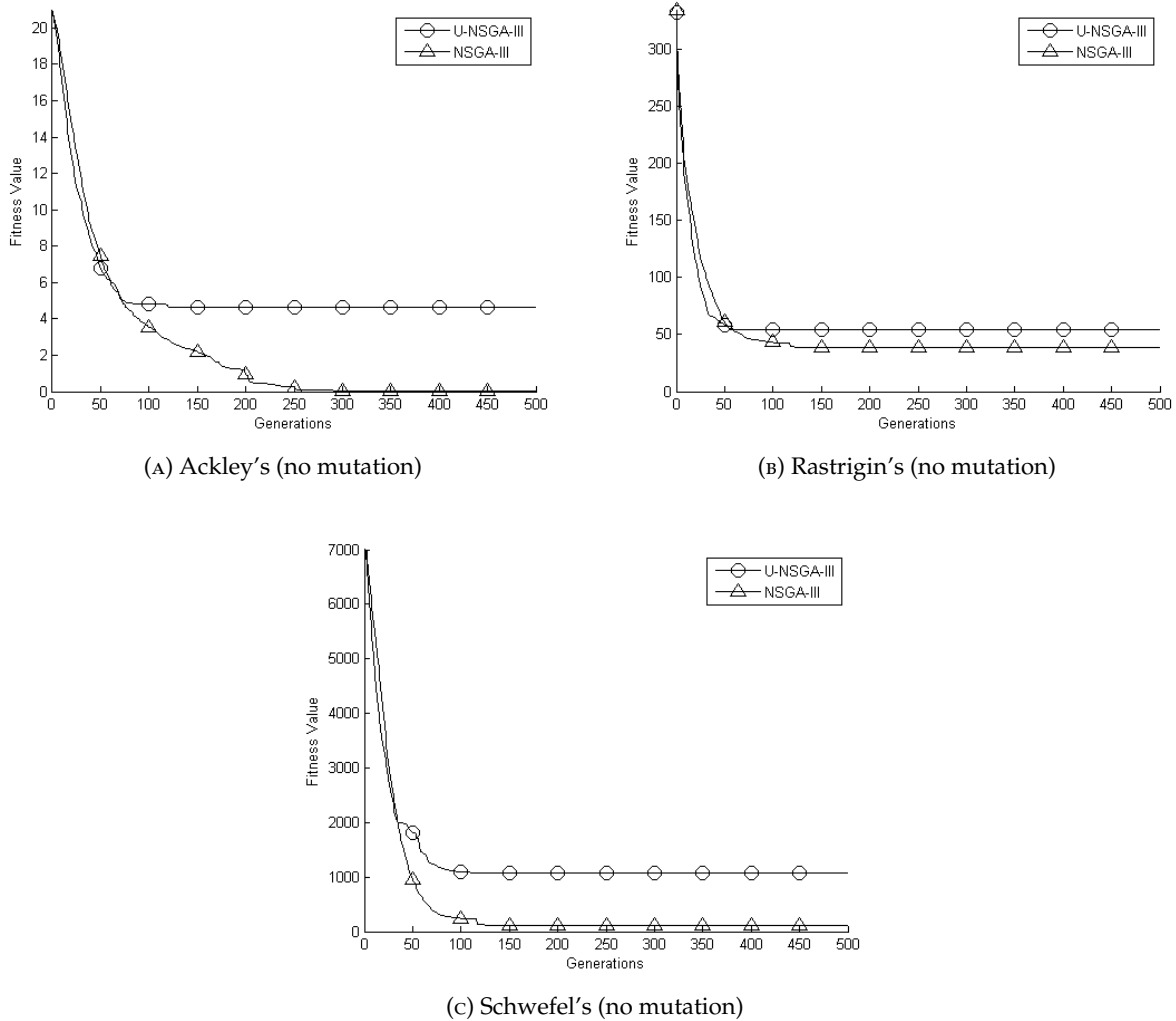


FIGURE 4.7: Small population size in multi-modal problems.

4.3 Summary

In this study we explored the performance of a multi-fold version of a recently proposed evolutionary many-objective optimization algorithm (NSGA-III), in which the population size is significantly larger than the number of pre-specified reference directions. This setting has never been tested before with NSGA-III. Through a careful set of computational optimizations, we have revealed a new and extended scope of the multi-fold NSGA-III. We have also proposed two hypotheses and discussed the correctness of each of them, both logically and empirically. Hypothesis 1 states that NSGA-III is slower than its selection-based counterparts when it comes to convergence. Hypothesis 2 states that NSGA-III has a higher ability to escape local optima and is less-dependent on mutation operators compared to its selection-based counterparts. A logically driven discussion of both hypotheses attributes their correctness mainly to the less greedy nature of NSGA-III due to the absence of selection. Finally, two carefully designed experiments have been conducted and the results obtained from both supported the correctness of our hypotheses.

The excellent performance of NSGA-III in many-objective problems demonstrated in the original two-part papers [4, 7] has now been aided in this study by showing the performance of a multi-fold NSGA-III approach. Thus, along with the original study, this study portrays a much wider scope of NSGA-III for solving different types of problems than the authors of original NSGA-III perceived.

Chapter 5

Towards Faster Convergence of Evolutionary Multi-Criterion Optimization Algorithms using Karush-Kuhn-Tucker Optimality Based Local Search

The following chapters address the convergence-diversity dilemma mentioned in Chapter 1. First, we try to deal with convergence in this chapter, then we move to the other problems in the remaining chapters. Here, we augment NSGA-III [4] with an efficient Local Search (LS) method that avoids most of the limitations mentioned previously (see Chapter 2). In Section 5.1, we summarize KKTPM computational procedure briefly.

Thereafter, we present and discuss the proposed hybrid algorithm in detail in Section 5.2. Section 5.3 presents our extensive simulation results, as well as some carefully chosen special cases showing the merits of our approach. Finally, we draw conclusions in Section 5.4.

5.1 A Brief Review of KKTPM NSGA-III

In this section, we first summarize the exact KKTPM computation procedure based on solution of an optimization task. Thereafter, we summarize a computationally fast yet approximate procedure which has been recently proposed.

5.1.1 Exact KKT Proximity Measure

As mentioned in Chapter 1, Deb et al. [18] defined an approximate KKT solution to compute a KKT proximity measure for any iterate (solution), $\mathbf{x}^k \in R^n$, for an M -objective optimization problem of the following type:

$$\begin{aligned} & \text{Minimize}_{(\mathbf{x})} \quad \{f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_M(\mathbf{x})\}, \\ & \text{Subject to} \quad g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, J. \end{aligned} \tag{5.1}$$

For a given iterate $\mathbf{x}^k$, they formulated an *achievement scalarization function* (ASF) optimization problem [82]:

$$\begin{aligned} & \text{Minimize}_{(\mathbf{x})} \quad \text{ASF}(\mathbf{x}, \mathbf{z}, \mathbf{w}) = \max_{m=1}^M \left(\frac{f_m(\mathbf{x}) - z_m}{w_m} \right), \\ & \text{Subject to} \quad g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, J. \end{aligned} \tag{5.2}$$

The reference point $\mathbf{z} \in R^M$ in the objective space can be considered a utopian point and each component of the weight vector $\mathbf{w} \in R^M$ is set for $\mathbf{x}^k$ as follows:

$$w_i = \frac{f_i(\mathbf{x}^k) - z_i}{\sqrt{\sum_{m=1}^M (f_m(\mathbf{x}^k) - z_m)^2}}. \quad (5.3)$$

Thereafter, the KKT proximity measure is computed by extending the procedure developed for single-objective problems elsewhere [79]. Since the ASF formulation makes the objective function non-differentiable, a smooth transformation was first made by introducing a slack variable x_{n+1} and reformulating the as follows:

$$\begin{aligned} & \text{Minimize}_{(\mathbf{x}, x_{n+1})} \quad F(\mathbf{x}, x_{n+1}) = x_{n+1}, \\ & \text{Subject to} \quad \left(\frac{f_i(\mathbf{x}) - z_i}{w_i^k} \right) - x_{n+1} \leq 0, \\ & \quad i = 1, \dots, M, \\ & \quad g_j(\mathbf{x}) \leq 0, \\ & \quad j = 1, 2, \dots, J. \end{aligned} \quad (5.4)$$

Now, the KKTPM optimization problem for the above smooth objective function $\mathbf{y} = (\mathbf{x}; x_{n+1})$ can be written as follows:

$$\begin{aligned}
& \text{Minimize}_{(\epsilon_k, x_{n+1}, \mathbf{u})} \quad \epsilon_k + \sum_{j=1}^J \left(u_{M+j} g_j(\mathbf{x}^k) \right)^2, \\
& \text{Subject to} \quad \left\| \nabla F(\mathbf{y}) + \sum_{j=1}^{M+J} u_j \nabla G_j(\mathbf{y}) \right\|^2 \leq \epsilon_k, \\
& \quad \sum_{j=1}^{M+J} u_j G_j(\mathbf{y}) \geq -\epsilon_k, \\
& \quad \left(\frac{f_j(\mathbf{x}) - z_j}{w_j^k} \right) - x_{n+1} \leq 0, \quad j = 1, \dots, M, \\
& \quad u_j \geq 0, \quad j = 1, 2, \dots, (M+J).
\end{aligned} \tag{5.5}$$

The constraints $G_j(\mathbf{y})$ are given below:

$$G_j(\mathbf{y}) = \left(\frac{f_j(\mathbf{x}) - z_j}{w_j^k} \right) - x_{n+1} \leq 0, \quad j = 1, \dots, M, \tag{5.6}$$

$$G_{M+j}(\mathbf{y}) = g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, J. \tag{5.7}$$

The value of ϵ_k^* at the optimal point of the above problem (Equation 5.5) corresponds to the exact KKT proximity measure. It is observed that for a feasible solution $\mathbf{x}^k$, $\epsilon_k^* \leq 1$; hence the exact KKTPM can be defined as follows:

$$\begin{aligned}
& \text{Exact KKT Proximity Measure}(\mathbf{x}^k) = \\
& \begin{cases} \epsilon_k^*, & \text{if } \mathbf{x}^k \text{ is feasible,} \\ 1 + \sum_{j=1}^J \left\langle g_j(\mathbf{x}^k) \right\rangle^2, & \text{otherwise.} \end{cases}
\end{aligned} \tag{5.8}$$

5.1.2 Approximate KKTPM Computation Method

There are three constraint sets to the optimization task in Equation 5.5. The first constraint requires computing the gradients of F and G functions and can be rewritten as:

$$\epsilon_k \geq \left\| \sum_{j=1}^M \frac{u_j}{w_j^k} \nabla f_j(\mathbf{x}^k) + \sum_{j=1}^J u_{M+j} \nabla g_j(\mathbf{x}^k) \right\|^2 + \left(1 - \sum_{j=1}^M u_j \right)^2. \quad (5.9)$$

The second constraint can be rewritten as follows:

$$\epsilon_k \geq - \sum_{j=1}^M u_j \left(\frac{f_j(\mathbf{x}^k) - z_j}{w_j^k} - x_{n+1} \right) - \sum_{j=1}^J u_{M+j} g_j(\mathbf{x}^k). \quad (5.10)$$

Since x_{n+1} is a slack variable, the third constraint will be satisfied by setting

$$x_{n+1} = \max_{j=1}^M \left(\frac{f_j(\mathbf{x}^k) - z_j}{w_j^k} \right). \quad (5.11)$$

A recent study [19] proposed a way to avoid the computationally expensive optimization procedure required to compute exact KKTPM and suggested several alternative approximate methods. The following paragraphs summarize these methods.

The first approximation (referred here as the ‘Direct’ method) ignored the second constraint given in Equation 5.10 and only the first (quadratic) constraint is used to find the KKTPM value. This is explained in Figure 5.1.

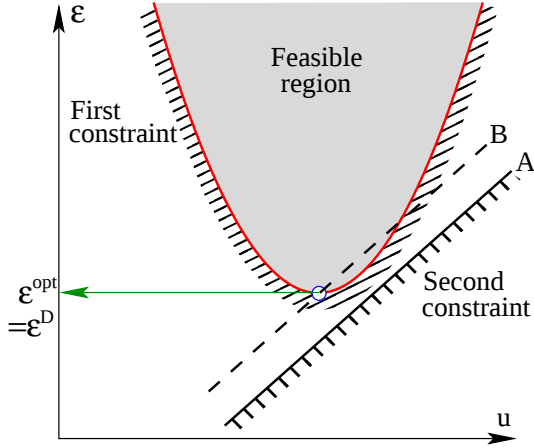


FIGURE 5.1: Second constraint governs at iterate $\mathbf{x}(k)$ which is not a KKT point.

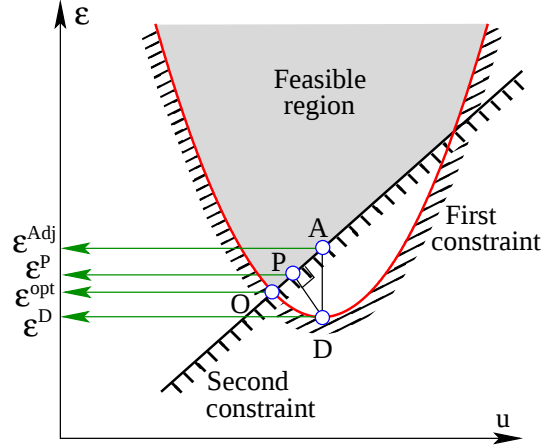


FIGURE 5.2: Second constraint governs at iterate $\mathbf{x}(k)$ which is not a KKT point.

In this case, the second constraint gets automatically satisfied at a point where the first constraint is feasible. The respective optimization problem is equivalent to solving a set of linear system of equations and the process gives rise to the following solution for ϵ_k^D :

$$\epsilon_k^D = 1 - \mathbf{1}_{M \times 1}^T \mathbf{u}_M^D - (\mathcal{G}^T \mathbf{u}_J^D)^2. \quad (5.12)$$

Since $u_m^D \geq 0$, $u_j^D \geq 0$, and $\mathcal{G}\mathcal{G}^T$ is a matrix with positive elements, it can be concluded that $\epsilon_k^D \leq 1$ for any feasible iterate $\mathbf{x}^k$. Deb et al. [19] has also observed that this scenario happens only when the following condition is true at $(\mathbf{u}_{M'}^D, \mathbf{u}_J^D)$ -vector:

$$1 - \mathbf{1}_{M \times 1}^T \mathbf{u}_M^D - (\mathcal{G}^T \mathbf{u}_J^D)^2 \geq -\mathcal{G}^T \mathbf{u}_J, \quad (5.13)$$

$$\text{or, } \mathbf{1}_{M \times 1}^T \mathbf{u}_M^D - (\mathcal{G}^T \mathbf{u}_J^D)(1 - \mathcal{G}^T \mathbf{u}_J^D) \leq 1. \quad (5.14)$$

Let us now get back to a more generic scenario (illustrated in Figure 5.2) in which the

condition 5.14 is not satisfied for an iterate $\mathbf{x}^k$. In this scenario, $\epsilon_k^* \neq \epsilon_k^D$ and in fact, $\epsilon_k^* > \epsilon_k^D$. Authors proposed approximate values of ϵ_k^* by using three computationally fast approaches, which we discuss in the following subsections.

5.1.2.1 Adjusted KKTPM Computation Method

From the direct solution $\mathbf{u}^D$ and corresponding ϵ_k^D (point 'D' in the Figure 5.2), we compute an *adjusted* point 'A' (marked in the figure), by simply computing the ϵ_k value from the second constraint boundary at $\mathbf{u} = \mathbf{u}^D$, as follows:

$$\epsilon_k^{Adj} = -\mathcal{G}^T \mathbf{u}_j^D. \quad (5.15)$$

5.1.2.2 Projected KKTPM Computation Method

Next, we consider another approximation method using $\mathbf{u}_j^D$. This time, we make a projection from the direct solution (point 'D') $(\mathbf{u}^D, \epsilon_k^D)$ on the second (linear) constraint boundary and obtain the *projected* KKTPM value (for point P), as follows:

$$\epsilon_k^P = \frac{\mathcal{G}^T (\epsilon_k^D \mathcal{G} - \mathbf{u}_j^D)}{1 + \mathcal{G}^T \mathcal{G}}. \quad (5.16)$$

5.1.2.3 Estimated KKTPM Computation Method

After calculating above approximate KKTPM values on many test problems and on a number of engineering design problems, the authors have suggested an aggregate KKTPM

value by averaging them and referring to it as the *estimated* KKTPM, as follows:

$$\epsilon^{est} = \frac{1}{3}(\epsilon_k^D + \epsilon_k^P + \epsilon_k^{Adj}). \quad (5.17)$$

5.2 Proposed KKTPM based Hybrid Local Search-EMO

The proposed hybrid approach involves an EMO procedure (NSGA-III), an identification process for choosing a population member for performing LS and a LS procedure.

5.2.1 Identification of a Population Member for Local Search

In a population based EMO procedure, non-dominated population members are *closest* to the Pareto-optimal set. However, without any knowledge of the true Pareto-optimal points, it is difficult to determine their relative proximity from the true Pareto-optimal front. Moreover, some solutions can stay non-dominated within the currently existing set of Pareto-optimal solutions in a population and can never be exposed being dominated unless one or more crucial but difficult-to-find Pareto-optimal points are discovered. In these situations, the relative KKTPM value of non-dominated solutions can play a crucial role.

For a set of non-dominated solutions in every generation of an EMO algorithm, KKTPM values can be calculated using the exact (Equation 5.8) or the approximate method (Equation 5.17) discussed above. Since a KKTPM value is found to be correlated to the

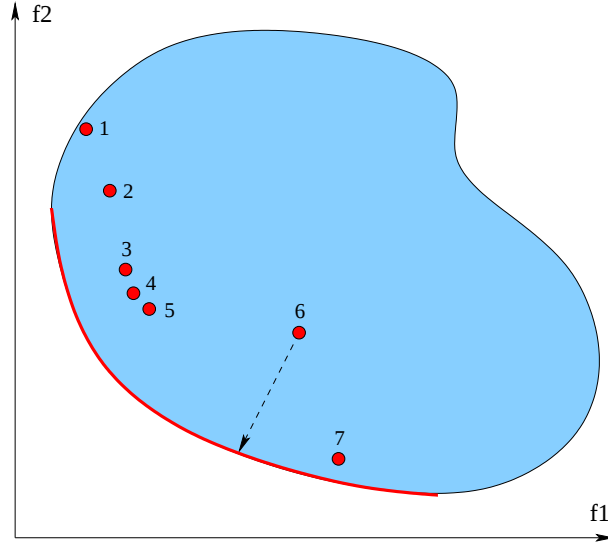


FIGURE 5.3: Identification of poorly converged non-dominated solution for LS.

distance from the efficient front [18], the non-dominated solution having the largest KK-TPM value is expected to have the worst convergence property and can be considered an ideal candidate for an improvement using a LS method. Figure 5.3 illustrates this aspect.

All seven points in the figure are non-dominated with respect to each other. Since the knowledge of the efficient front is not available, it is difficult to evaluate the relative closeness of each non-dominated solution from the efficient front. The usual crowding information (such as crowding distance used in NSGA-II) or other similar niching metrics do not provide any such closeness information. The crowding distance may find point 4 to be the worst, but in terms of closeness, it may not be the worst of all points. KKTPM value provides us with closeness information which can be exploited to identify the worst non-dominated point (point 6 in the figure) in terms of convergence.

5.2.2 Local Search Procedure

After a poorly converged population member $\mathbf{x}^w$ is identified using KKTPM, the solution is improved by applying an LS method. In the presence of multiple conflicting objectives, we would require a scalarization method to execute a LS. The achievement scalarizing function (ASF) [82] is one scalarization method that has been popularly used in the recent past in EMO studies. It requires two entities: (i) a reference point $\mathbf{z}^r$, and (ii) a weight vector $\mathbf{w}$:

$$\begin{aligned} \underset{\mathbf{x}}{\text{Minimize}} \quad & ASF(\mathbf{x}, \mathbf{z}^r, \mathbf{w}) = \max_{i=1}^M \left(\frac{\tilde{f}_i(\mathbf{x}) - u_i}{w_i} \right), \\ \text{subject to} \quad & g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, J. \end{aligned} \tag{5.18}$$

The use of ASF benefits from being used with NSGA-III, which already uses a set of predefined reference directions in its operation. Let us say that the solution $\mathbf{x}^w$ is associated with reference direction $\mathbf{w}$. This reference direction vector can be used in Equation 5.18 directly. Also the ASF function can directly use the ideal point and intercepts found so far during the course of NSGA-III operation, to transform the objectives vector f to the normalized objectives vector $\tilde{f}$. The utopian point ($\mathbf{u}$) is calculated by subtracting a very small value (typically 10^{-4}) from all the coordinates of the ideal point $\mathbf{z}^r$, as shown in Figure 5.4.

To minimize the ASF, we use $\mathbf{x}^w$ as the starting point and employ Matlab's[®] `fmincon()` routine as the LS algorithm.

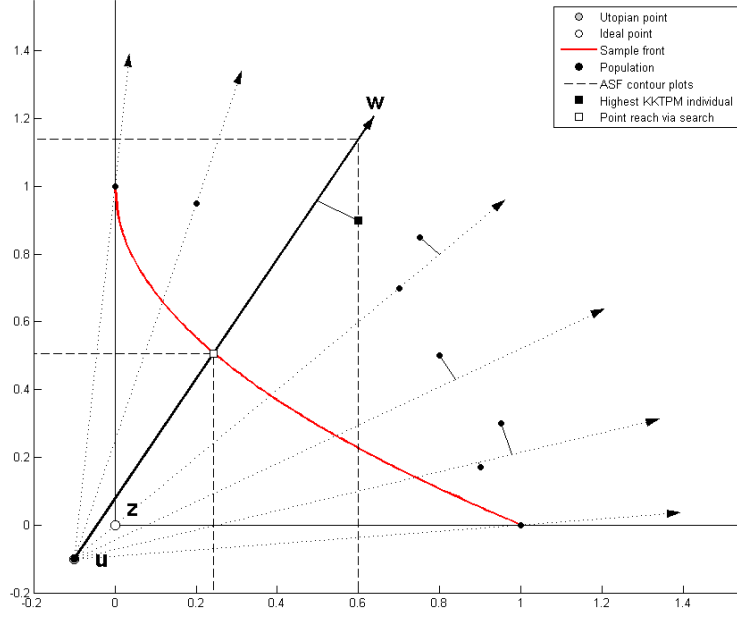


FIGURE 5.4: ASF-based LS performed in the normalized objective space.

5.2.3 Overall Proposed Algorithm

We are now ready to present the overall hybrid LS based EMO procedure. The whole procedure is described in Algorithm 5.

ALGORITHM 5 KKTPM-based LS procedure.

Input: Population (P), ideal point (D), intercepts (I), utopian point (U), maximum number of function evaluations (FE_MAX)

Output: Population after local search P'

- 1: $F \leftarrow \text{getFeasible}(P)$
 - 2: $ind_s \leftarrow \mathbf{x} \mid \mathbf{x} \in F \ \& \ KKTPM(\mathbf{x}) \geq KKTPM(\mathbf{y}) \ \forall \ \mathbf{y} \in F$
 $\% \ ind_s$: initial solution for local search
 - 3: $ind_f \leftarrow \text{localSearch}(ind_s, ind_s.dir, D, I, U, FE_MAX)$
 $\% \ ind_f$: solution after local search
 - 4: $P' \leftarrow \text{replace}(P, ind_f)$
-

The idea is to pick the worst feasible individual (having the largest KKTPM value) in every generation. If this solution is infeasible, LS is postponed until a feasible solution is found. If the solution is feasible, this means that it has already been associated with a specific reference direction. Using this individual as a starting point, an ASF-based LS is

then performed along the direction to which it is associated. Finally, the original solution is replaced by the individual resulting from the LS.

5.2.4 Frequency of Local Search

It may be tempting to perform the above LS for every individual in every generation, but this is not a good idea considering the overall computational burden. A single LS operation is a complete single-objective optimization process (performed using the `fmincon()` routine). The number of function evaluations needed to solve an ASF problem is unknown beforehand. It depends on the starting point, the LS algorithm and the problem itself. To control the number of Solution Evaluations (SEs), in our implementation, a *maximum* limit of 500 solution evaluations per LS is allowed ($SE_{LS}=500$). LS can also be performed once every T generations. Although this parameter needs tuning, our initial investigation showed that a value of $T = 1$ works better on difficult problems while $T = 5$ suite simpler problems more.

5.3 Results

In this section, we present simulation results on a number of test problems and a few engineering design problems. But before we do that, we highlight the importance of using the proposed LS based hybrid approach by demonstrating its use on a special problem. In all simulations, we use the SBX recombination operator [32] with $p_c = 0.9$ and $\eta_c = 30$ and the polynomial mutation operator [5] with $p_m = 1/n$ (where n is the number of variables) and $\eta_m = 20$. All other parameters are shown in Table 5.1.

TABLE 5.1: Parameters - Columns represent – left-to-right – problem name, population size, number of function evaluations, frequency of LS (per generation), maximum allowed number of function evaluations per one LS, number of generations with LS and number of generations without LS

Problem	Popsiz (N)	Frequen (T)	$FE_{\max}$ for LS (FE_{LS})	# Gen. for Hybrid EMO	Total SE	# Gen. for Orig. EMO
Var-Dens	40	5	200	200	14490	334
ZDT1	40	5	200	200	16053	400
ZDT2	40	5	200	200	15017	374
ZDT3	40	5	200	200	15193	378
ZDT4	48	5	500	300	32776	681
ZDT6	40	5	200	200	11595	288
TNK	20	1	200	200	15705	784
BNH	40	1	500	300	30219	754
SRN	40	1	500	300	27986	698
OSY	40	1	500	300	43030	1074
DTLZ1(3)	92	5	500	500	68159	739
DTLZ1(10)	276	5	500	500	175157	633
DTLZ2(3)	92	5	500	400	49917	541
DTLZ2(10)	276	5	500	400	143271	518
DTLZ5(3)	92	5	500	400	52645	571
WELD	40	1	500	300	30499	761
CAR	92	1	500	500	85792	931

Population size N , frequency of local search T , solution evaluations for each local search SE_{LS} and number of generations for the hybrid NSGA-III approach are set based on standard practices and complexity of each problem. After the hybrid approach is run for prescribed number of generations with local search, the total SE (TSE) is recorded for each run and the median value is presented in Table 5.1. TSE is then set for the original NSGA-III procedure and the number of generations needed to spend an identical TSE is computed and tabulated in the last column. This allows us to compare our proposed hybrid approach with the original NSGA-III for an identical SE count.

5.3.1 A Variable-Density Problem

Consider the following n -dimensional variable density (Var-Dens) problem:

$$\begin{aligned}
& \underset{\mathbf{x}}{\text{minimize}} && f_1(\mathbf{x}) = x_1^\alpha, \\
& \underset{\mathbf{x}}{\text{minimize}} && f_2(\mathbf{x}) = g(\mathbf{x}) \left(\frac{\cos(\pi\beta f_1(\mathbf{x})) - (3.5f_1(\mathbf{x}) - 1)^3 + 16.625}{18.625} \right), \\
& \text{subject to} && g(\mathbf{x}) = 1 + \frac{1}{n-1} \sum_{i=2}^n x_i, \\
& && 0 \leq x_i \leq 1, \quad i = 1, \dots, n.
\end{aligned} \tag{5.19}$$

α controls the degree of sparseness as we move across the objective space, while β determines the number of *fallacious* Pareto points (regions) that will not be removed until some dominating point from a less sparse region is found. In our implementation, α is set to 0.05 and β is set to 5. A careful analysis of the problem reveals that this problem introduces a biased density of points towards large f_1 objective values. Figure 5.5 shows the relative density of 10,000 objective vectors created randomly in the two-variable search space. It is clear that points near small f_1 are rare. The problem has two disjointed efficient fronts: AB and CD. Due to the varying density of solutions in the objective space, it may be difficult to find the part AB, whereas it is expected that an EMO algorithm will find the part CD relatively easily.

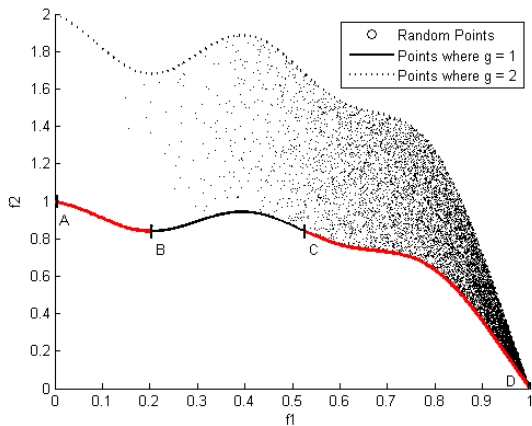


FIGURE 5.5: Pareto-optimal front and relative density of solutions for the varying density problem.

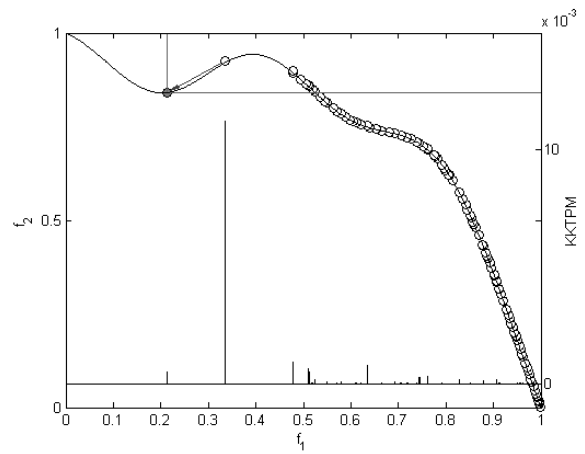


FIGURE 5.6: Proposed LS replaces non-Pareto-optimal solution with a true Pareto-optimal solution.

Figure 5.6 shows a typical performance of the proposed hybrid algorithm with 100 population members. Non-dominated solutions after 200 generations are shown in the figure and they seem to fall on the true Pareto-optimal front.

It is clear that the part AB of the front is undiscovered and this allows a few non-Pareto-optimal solutions to tag along with the other easy part of the Pareto-optimal front. However, when KKTPM values are found for all these non-dominated solutions, it is observed that one of the non-Pareto-optimal point has the worst KKTPM value (shown with bars – larger value signify worse convergence). Once this point is identified and an LS is performed starting at it, the point gets replaced by a Pareto-optimal solution on the part AB of the Pareto front. The discovery of this new Pareto-optimal point by LS achieves two desired goals:

1. It helps improve the convergence property of the original set of non-dominated solutions, and
2. It also helps eliminate some non-Pareto-optimal solutions that are still present in the non-dominated set.

Both of the above properties help make the overall search procedure faster and more convergent. It is not clear how else such desired goals can be achieved by using existing methods.

5.3.2 Bi-Objective Test Problems

We now present results of our proposed hybrid procedure applied to a number of bi-objective constrained and unconstrained problems. For all problems, we compare our

results to the original NSGA-III for an identical SE count. Finally, all our aggregate results are calculated for 31 independent runs, starting from different initial populations.

5.3.2.1 Specific Test Problems

First, we consider results on three specific test problems to illustrate in detail the usefulness of our hybrid NSGA-III procedure.

We consider the constrained problem OSY [5]. Figure 5.7 presents the variation of the smallest, median and the worst KKTPM values for original NSGA-III and hybrid NSGA-III procedures on problem OSY [5].

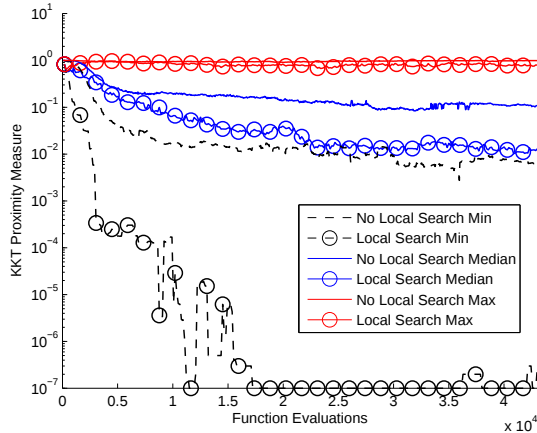


FIGURE 5.7: KKTPM comparison with and without LS on problem OSY. The lines with circles are from the LS based method.

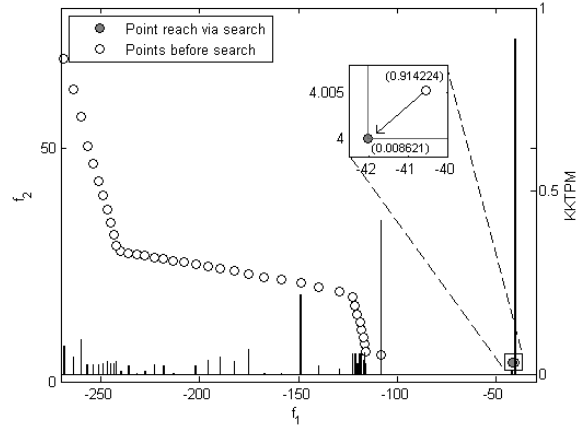


FIGURE 5.8: Effect of LS on problem OSY.

x -axis represents SE count in both cases. Since hybrid NSGA-III spends additional SEs on every LS operation, SE count is used as a basis for comparison instead of the usual number of generations. In all figures, circles mark our hybrid approach. Since circled lines show smaller KKTPM values than respective non-circle lines, we can safely conclude that the use of LS has significantly improved KKTPM values, thereby indicating a better convergence property of the proposed hybrid procedure.

Figure 5.8 shows the obtained non-dominated front for a typical run after 1,072 generations using a population of 40.

The figure also plots the KKTPM value of each of the non-dominated point using a bar. It can be observed that although small f_1 points have small KKTPM values, the largest f_1 point has a KKTPM value of almost one (0.9142). It is important to note that this extreme point stays non-dominated with the rest of well-converged trade-off points, but the fact that it has a very high KKTPM value indicates that this point may not be a part of the true Pareto-optimal front. The inset figure shows that when this poorly converged point is locally searched using the ASF based LS presented in Section 5.2.2, a new and well-converged point with a KKTPM value of only 0.0086 is found. Although not visually obvious, this new point is very close to the true Pareto-optimal front, as evident from its small KKTPM value. Thus, without the use of KKTPM information, it will be difficult to identify such poorly converged points, which can then be improved using the proposed LS procedure. Identification of such poorly converged points independently and their modification to find better converged solutions through genetic operations (such as selection, recombination and mutation) may require a very large number of generations, thereby making the convergence process too slow. A direct identification and fixation of such solutions using KKTPM values and LS operations makes the whole process computationally fast, as evident from Figure 5.7.

Next, we consider the BNH problem with a population of size 40. Figure 5.9 shows the variation of KKTPM for NSGA-III with and without the LS procedure for identical SE counts.

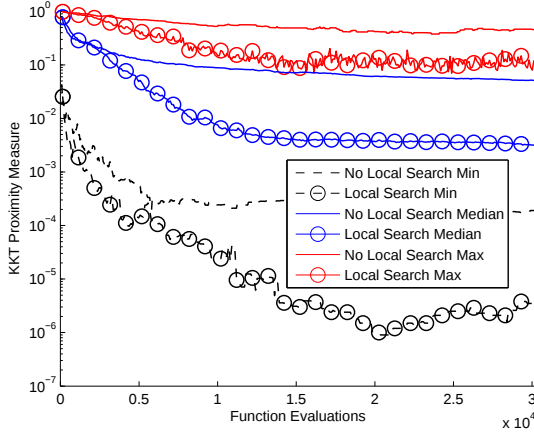


FIGURE 5.9: KKT comparison with and without LS on BNH.

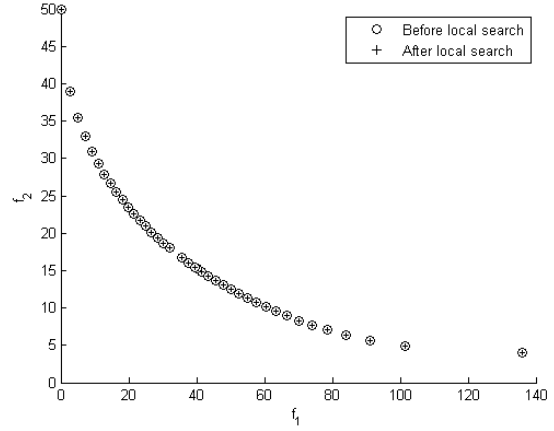


FIGURE 5.10: Effect of LS on BNH objective space.

The figure shows that a remarkable improvement in convergence behavior is obtained with the LS procedure. To illustrate the effect of LS, we plot the variation of two variables (x_1 and x_2) for non-dominated points after 250 generations in Figure 5.11.

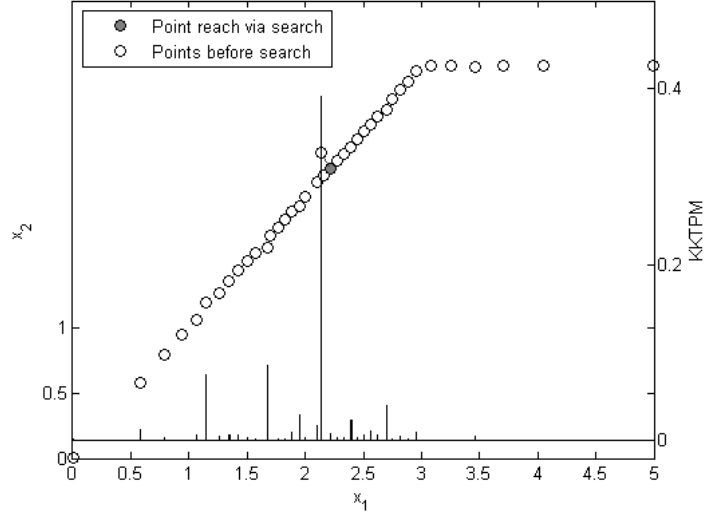


FIGURE 5.11: Effect of LS on BNH variable space.

The KKT values are also shown for each point. It can be observed that variables follow a pattern, a matter which has been observed in many real-world problems as well and has the potential of revealing innovative solution principles [105]. However, for this specific

case, there is one point – although still non-dominated with respect to the rest of the non-dominated set – seems to be out of sync with the observed pattern (see Figure 5.10).

Interestingly, the deviation from the pattern from other non-dominated solutions cannot be observed visually from the objective space plot. However, The KKTPM value for this point is computed to be large (0.39057), indicating a poor convergence property of this point. When this solution is identified and locally searched using our ASF scalarization procedure, the point gets improved and agrees with the observed pattern. The KKTPM value of this improved point becomes 0.00118 after being 0.39057 before LS. Interestingly, the two sets of points (before and after LS) seem to have very similar distributions in the objective space, which means that in this case an investigation of the variable space was necessary to reveal poorly converged non-dominated points. Irrespective of poor convergence of solutions in variable or objective spaces, KKTPM is capable of identifying them for a subsequent LS to improve their convergence behavior.

The third specific problem is ZDT4. Generally, our KKTPM/ASF-based hybrid approach takes the least converged individuals – with the highest KKTPM value – and pull them towards the front along their already pre-defined reference directions. This is expected to enhance convergence significantly. To show this effect we compared our approach to the original NSGA-III on ZDT4 test problem [99]. As mentioned before, ZDT4 is a hard problem having 21^9 local Pareto-optimal fronts. It requires at least 100 individuals with a relatively high number of generation (≥ 300 generations) for NSGA-III to converge to the true Pareto front [18]. For the sake of comparison, we use a relatively small population of only 48 individuals as an additional hurdle against the optimization algorithm. Out of

31 independent runs for each algorithm, the run having the median hypervolume (HV) is plotted in Figure 5.12.

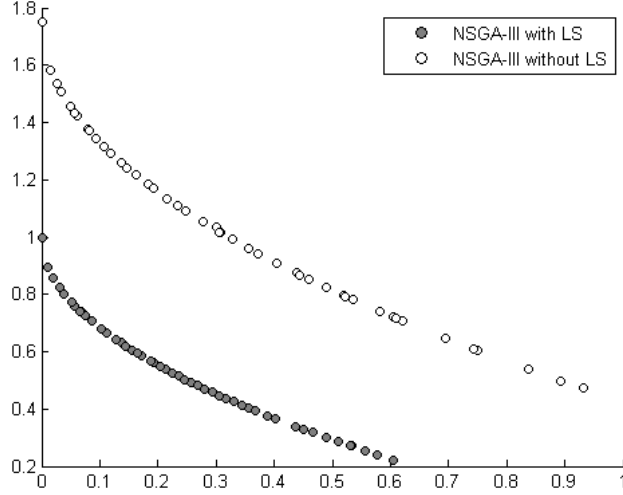


FIGURE 5.12: NSGA-III with and without LS on ZDT4.

Both algorithms use the same number of SEs per run. It is clear from the figure that the original NSGA-III is not able to converge to the Pareto front, whereas our hybrid NSGA-III is, but with worse diversity.

For a multi-modal problem, the value of KKTPM is dependent on how far an individual is from its closest *local* Pareto-optimal front. This is because every local optimum is also a KKT point and will have a zero KKTPM value. In other words, in a problem like ZDT4 having a large number of local Pareto-optimal fronts, KKTPM cannot really tell how far a point is from the global Pareto front. It can rather tell how far a point is from a local Pareto front. But since we pick the worst KKTPM point from the best non-dominated front in a population, it is likely that LS will eventually contribute to convergence. The behavior of ZDT4 shows what we call the "*leaping*" effect. The term "*leap*" here refers to the big jump one point makes so that it dominates the rest of the population. It is highly unlikely

to experience *Leaping* with only standard genetic operators (selection, recombination and mutation). However, using our selective LS procedure, we can increase its probability. In order to investigate *leaping* in our proposed approach, we recorded the frequency of LS operations that produced a solution which dominates the entire parent population. One such leap is illustrated in Figure 5.13a. The histogram plotted in Figure 5.13b shows the frequency of leaps across generations. It is clear that leaps are more frequent at the beginning and they decrease rapidly as generations proceed. On one side, *leaping* helps convergence, but on the other side, this may negatively affect diversity as shown in Figure 5.12.

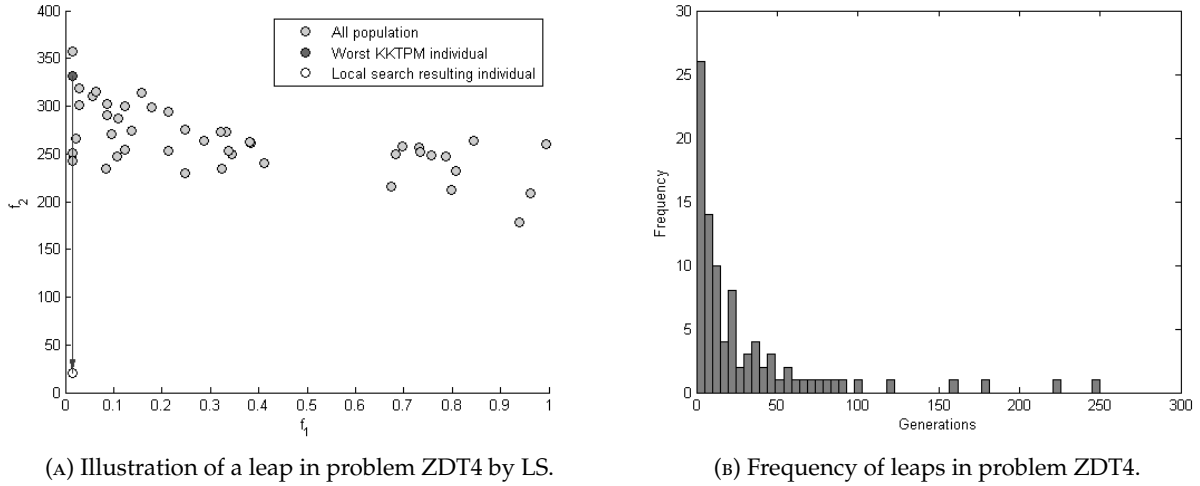


FIGURE 5.13: Leaps enable faster convergence on ZDT4

This indicates that our LS procedure using Matlab's[®] `fmincon()` routine is able to avoid many local Pareto-optimal fronts, resulting in better convergence (see Figure 5.12).

The few previous examples show a microscopic view of how our hybrid approach works and how it positively affects convergence. In the remaining part of this chapter we conduct extensive simulations on a wide range of bi-objective problems, in order to see the bigger

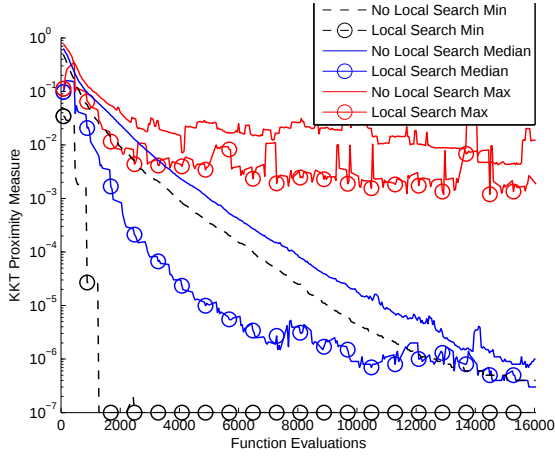
picture. In our simulations we run each algorithm 31 times on each test problem from different initial populations. For each algorithm, the minimum, median and maximum individuals in terms of KKTPM are selected for each generation across all runs. Each plot compares these three metric values from the first to the last generation.

Our bi-objective simulations include two sets of test problems. The first set has six unconstrained test problems: which are ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6 [99] and the proposed *variable-density* problem. Figures 5.14a to 5.14e present KKTPM values for ZDT1 to ZDT6 problems for NSGA-III and hybrid NSGA-III algorithms. Hybrid NSGA-III results are shown using circle-marked lines.

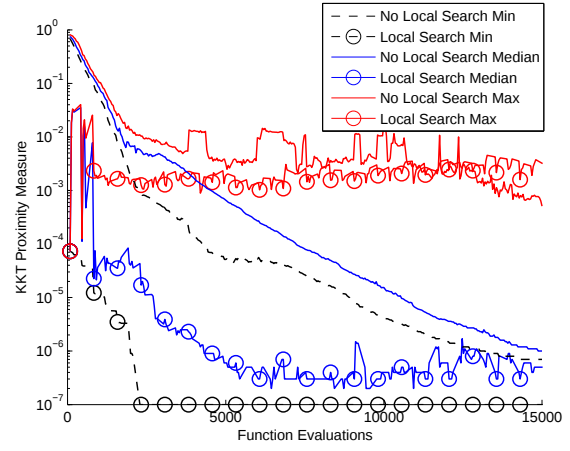
Figures 5.14f, 5.15a, and 5.15b show similar plots for *variable-density* problem, SRN and TNK, respectively. It is clear from these figures that for all problems the hybrid NSGA-III has achieved much smaller KKTPM values with identical number of SEs.

Figure 5.16a shows the obtained non-dominated set of solutions using both original and hybrid NSGA-III on ZDT1. The figure shows clearly that the hybrid approach is able to converge better than the original NSGA-III approach. Figure 5.16b shows a similar behavior on ZDT6.

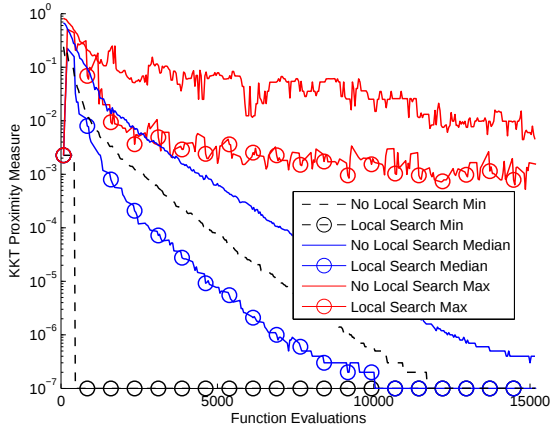
Figure 5.14f needs a more detailed explanation. In this case, LS was able to reach the hard-to-reach section of the Pareto-optimal front in most of our simulations. This is the reason behind the clear improvement in the highest KKTPM value shown in the figure. On the other hand, there is no visible improvement in the median KKTPM. Since the relatively easy-to-reach part of the Pareto-optimal front is bigger than the harder-to-reach part, the median KKTPM comes from the easy-to-reach part and hence there is not



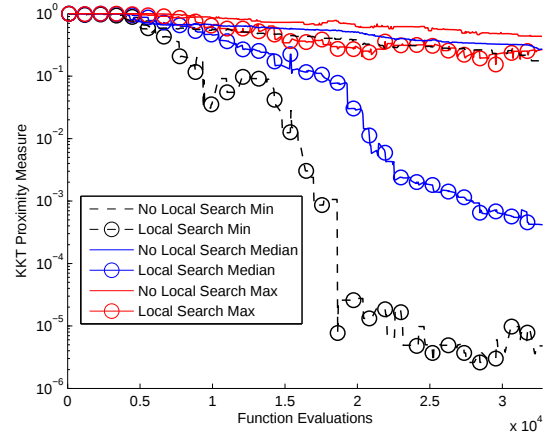
(A) KKTPM for ZDT1.



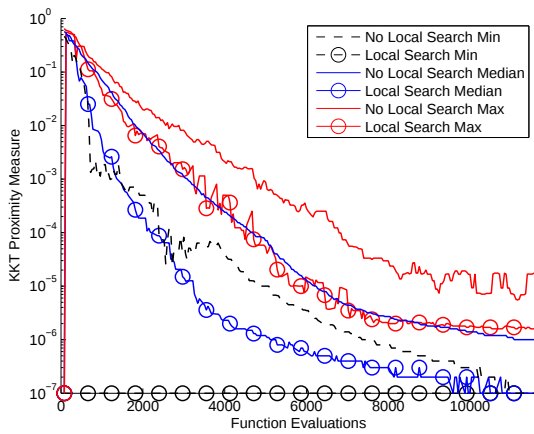
(B) KKTPM for ZDT2.



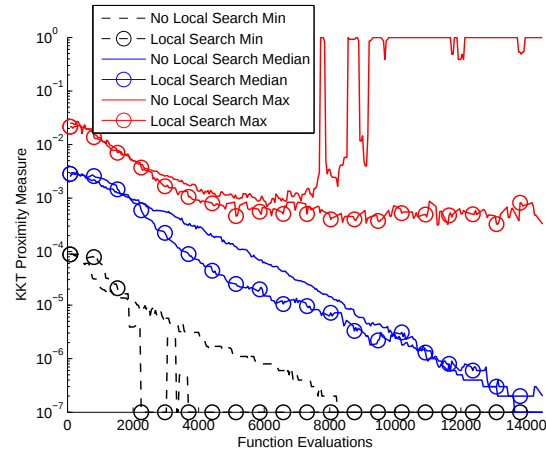
(C) KKTPM for ZDT3.



(D) KKTPM for ZDT4.

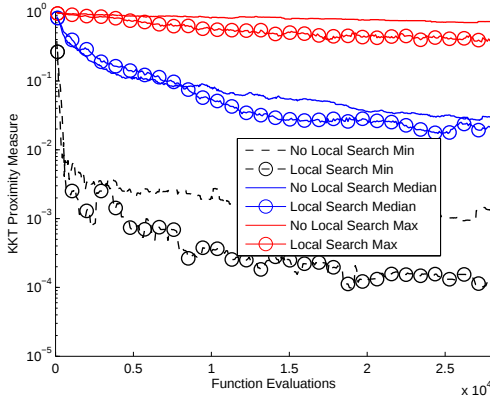


(E) KKTPM for ZDT6.

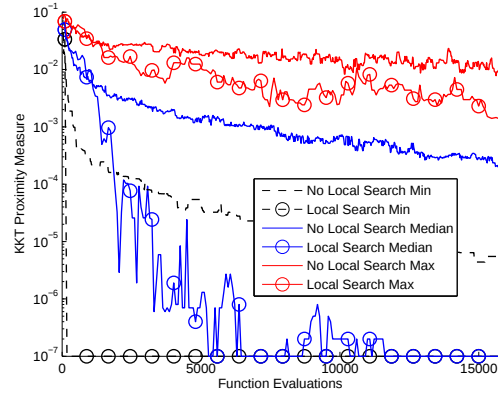


(F) KKTPM for *variable-density* problem.

FIGURE 5.14: Effect of LS on bi-objective unconstrained test problems. The hybrid approach shows a better convergence property.

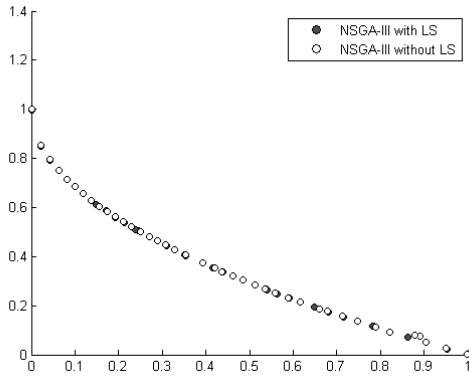


(A) KKTPM for SRN

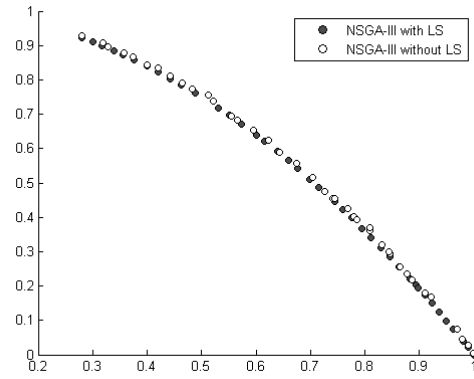


(B) KKTPM for TNK

FIGURE 5.15: LS effect on bi-objective constrained test problems SRN and TNK. The hybrid approach shows a better convergence property.



(A) ZDT1 front

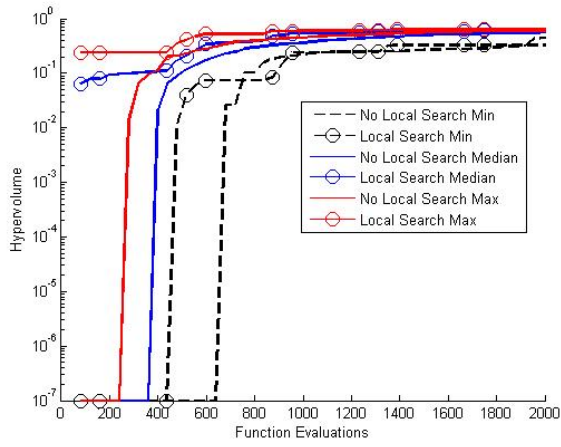


(B) ZDT6 front

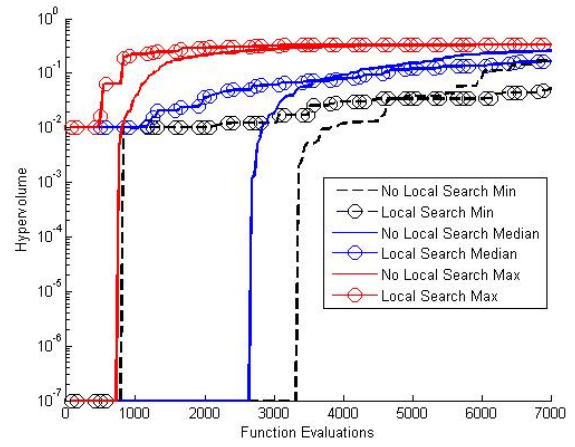
FIGURE 5.16: Final fronts obtained for ZDT1 and ZDT6 with and without LS. The hybrid approach with LS converges better.

much difference in the median KKTPM values between NSGA-III and hybrid NSGA-III. However, significant improvements in best and worst KKTPM values are obvious in the hybrid NSGA-III approach.

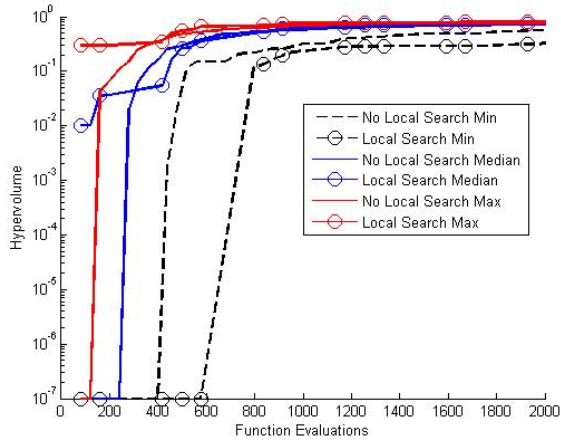
Similar observations can be made for bi-objective constrained test problems shown in Figure 5.15. Just like the previously discussed ZDT problems, OSY and BNH, in these two problems a clear improvement in all KKTPM values is visible from the figures.



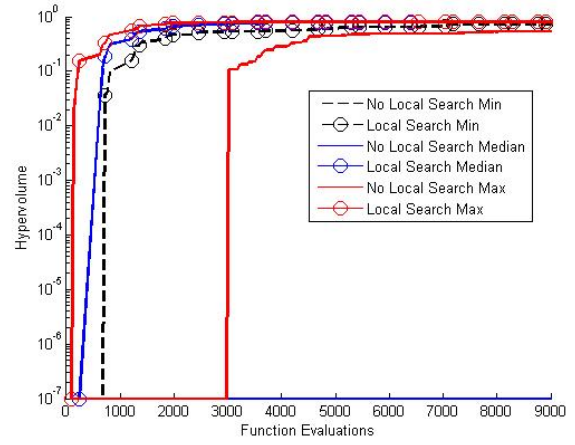
(A) ZDT1 HV



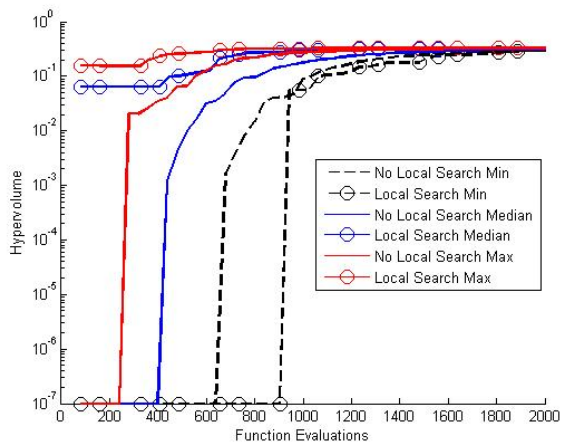
(B) ZDT2 HV



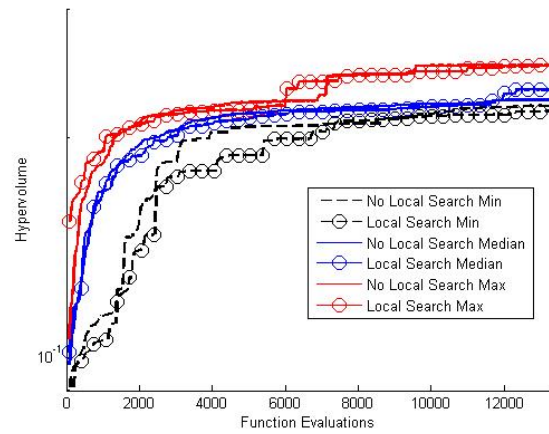
(C) ZDT3 HV



(D) ZDT4 HV



(E) ZDT6 HV



(F) VarDens HV

FIGURE 5.17: Hypervolumes compared with and without LS for bi-objective unconstrained problems.

KKTPM is expected to indicate and, if used in an optimization algorithm, improve the convergence characteristics of a solution. Thus, the proposed hybrid approach is not expected to improve the diversity aspect of an EMO. But to investigate if an emphasis on convergence deteriorates the diversity aspect, we compute the hypervolume (HV) of the obtained non-dominated set of solutions at every generation and plot them for all the above unconstrained problems using both original and hybrid NSGA-III in Figure 5.17.

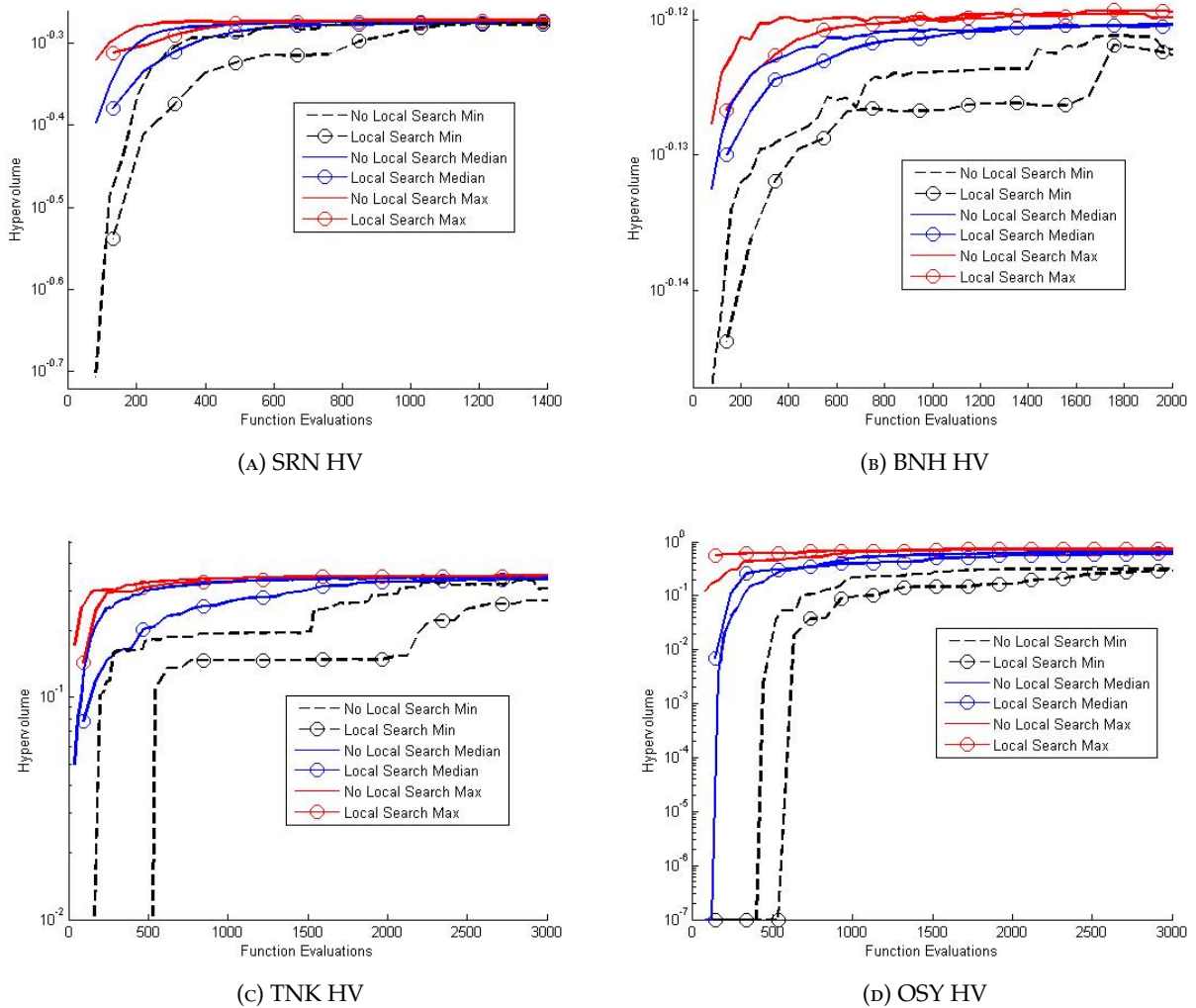


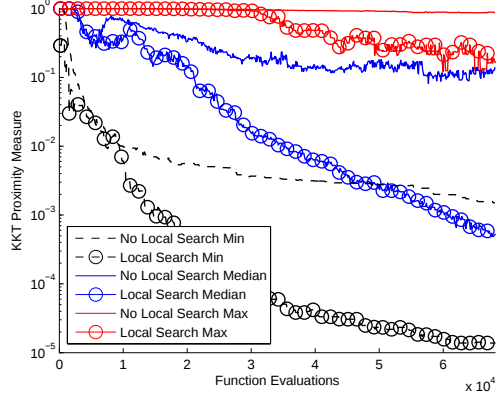
FIGURE 5.18: Hypervolumes compared with and without LS for bi-objective constrained problems. In most cases, hypervolume is better for the hybrid approach.

All hypervolumes are calculated using a reference point that is equal to 1.1 times the

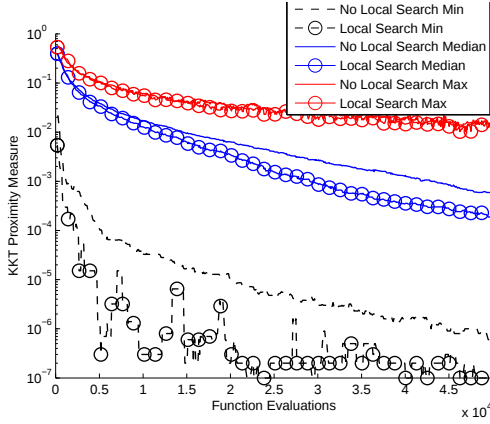
known nadir point of each test problem. The results show that with simple problems both approaches are comparable. Some observations can be made as problems gets more challenging. In some problems, the reader can notice that HV values of the hybrid approach shoots up early in the optimization process. This early improvement can be attributed to the initial fruitful LS operations. In most cases, original NSGA-III will be able to catch up and in some cases even achieve better HV values given enough time (generations). This can happen because of the ability of original NSGA-III to achieve better diversity in some problems. But mostly, this is because, in easy problems, early LS operations are less useful. They spend too much SEs compared to the amount of improvement they achieve (if any). Finally, on difficult problems (e.g. ZDT4) where the original NSGA-III fails to reach the Pareto front, our hybrid approach remains superior. Figure 5.18 shows a similar comparison of HV for constrained problems.

These figures amply indicate that the two approaches work differently from start to end, ultimately achieving a similar HV, but our proposed hybrid approach finds points that are much better converged than the original NSGA-III approach.

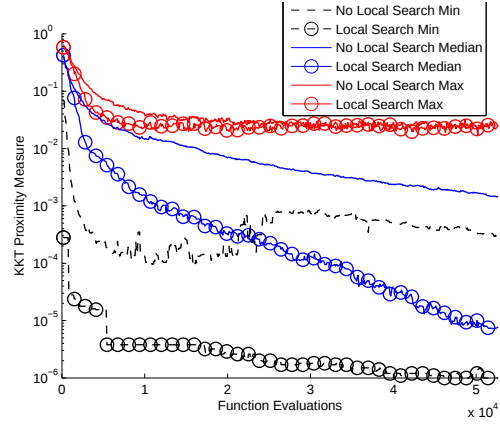
Identification of worst-converged non-dominated solution and its improvement using an ASF-based LS procedure demonstrated in ZDT and standard constrained test problems have indicated a faster convergence of the hybrid NSGA-III approach. It is important to highlight that there is no other way to differentiate one non-dominated solution from another in terms of their convergence characteristics and use of KKTPM provides us with this opportunity to apply a special action on less-converged solutions for an improvement of the overall algorithm.



(A) KKTTPM for DTLZ1



(B) KKTTPM for DTLZ2



(C) KKTTPM for DTLZ5

FIGURE 5.19: Effect of LS on three-objective DTLZ problems demonstrating better convergence property.

5.3.2.2 Three and Many-objective DTLZ Problems

We now consider three and many-objective DTLZ test problems [104]. Figures 5.19a to 5.19c show the variation of KKTTPM values on three-objective DTLZ1, DTLZ2, and DTLZ5 problems. A remarkable improvement in the convergence speed is observed by hybrid NSGA-III approach.

From [4], it is known that for a population of 92 solutions, it takes NSGA-III about 400 generations to reach close enough to the true Pareto-optimal front with an appropriate

distribution. After 400 generation it is hard if not impossible to detect any further improvement just by naked eye. Here we limit our number of generations to 323 generations (without LS) in order to compare the results of our two approaches during the course of the optimization process. For our hybrid approach we used only 250 generations, thus having an equal number of SEs for both approaches. Figures 5.20a and 5.20b show the final DTLZ1 fronts reached by each approach at the aforementioned generations.

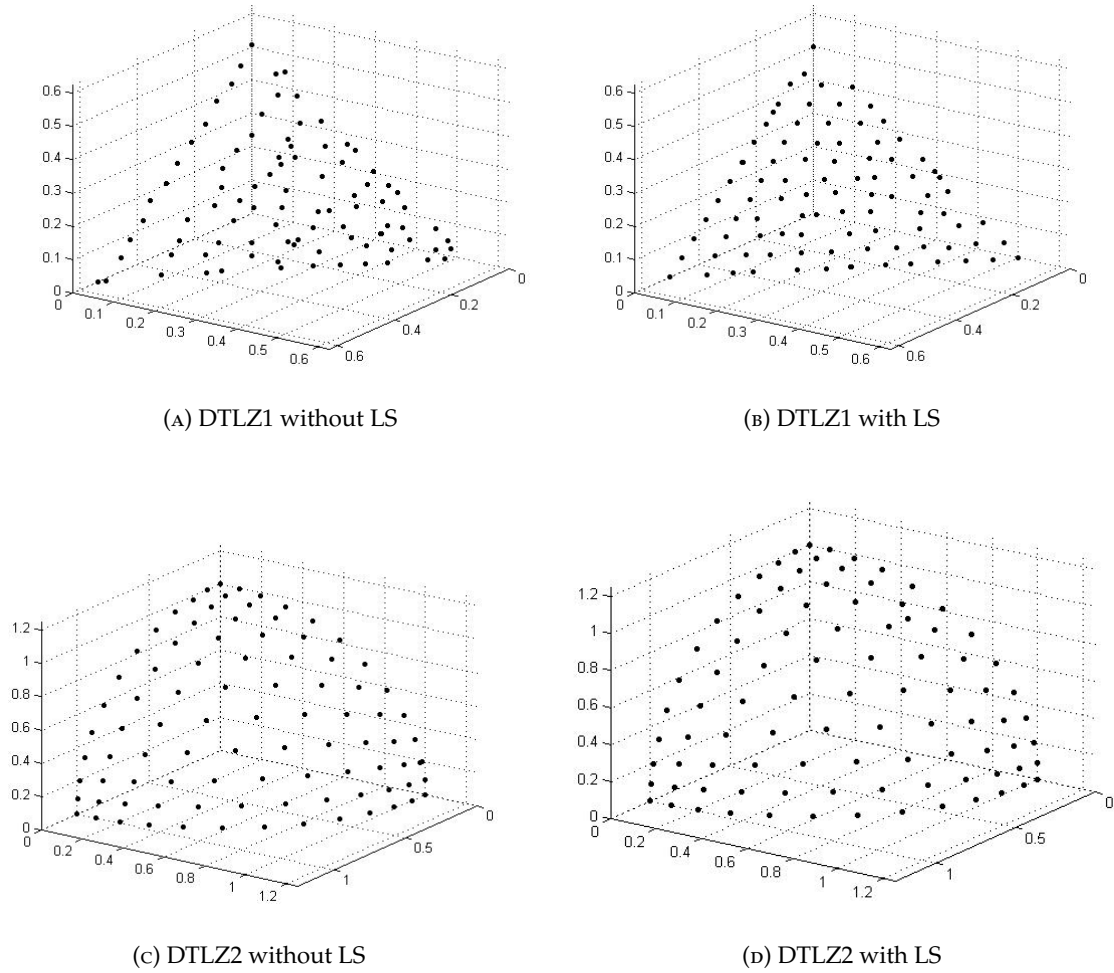


FIGURE 5.20: Final fronts of DTLZ1 and DTLZ2, with (right) and without (left) local search, on three objectives obtained with a limited FE.

It is clear that although both the two approaches did not fully converge (because of the limited number of SEs allowed, a better final front is reached using the hybrid approach.

Similar figures (5.20c and 5.20d) for DTLZ2 are shown but obviously the problem is too easy to show any superiority of one approach over the other. HV values of both DTLZ1 and DTLZ2 over generations are compared in Figure 5.21.

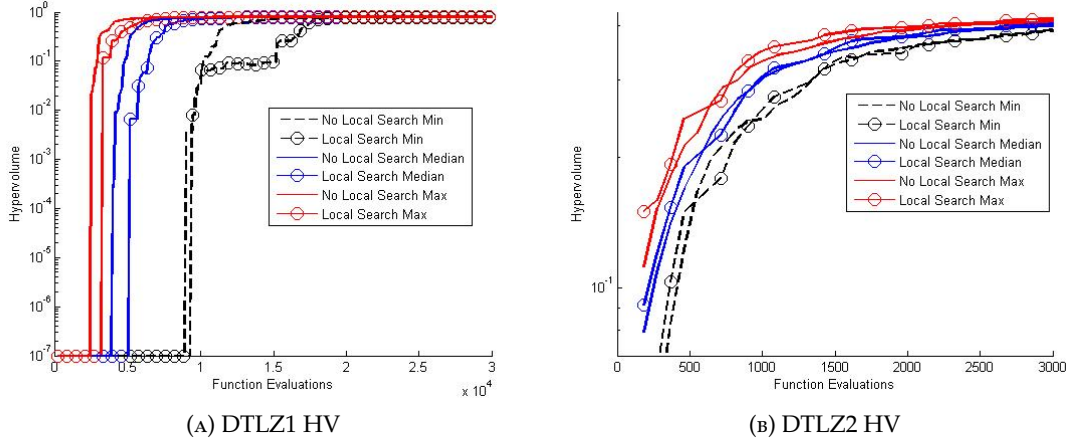


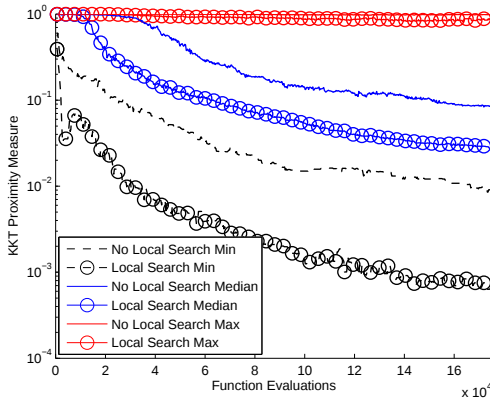
FIGURE 5.21: Comparable hypervolume for DTLZ1 and DTLZ2 with and without LS on three objective test problems.

To demonstrate the advantage of hybrid NSGA-III approach on many-objective problems, we consider 10-objective DTLZ1 and DTLZ2 problems. Figures 5.22a and 5.22b show the variation of best, median and worst KKTPM values using both original and hybrid NSGA-III approaches.

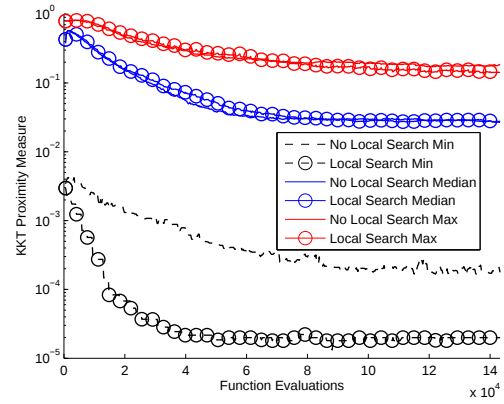
The performance of hybrid NSGA-III is significantly better on DTLZ1, but since DTLZ2 is a relatively easy problem to solve, the performance of the hybrid NSGA-III is marginally better. In general, our hybrid approach is able to improve the best KKTPM values significantly.

5.3.2.3 Engineering Design Problems

Finally, we include two engineering design problems: welded-beam design (WELD) [49] and car-side-impact (CAR) [7] problems. In Figures 5.23a and 5.23b the minimum KKTPM



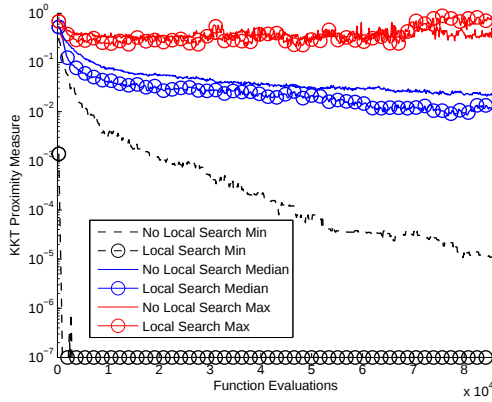
(A) KKTPM for DTLZ1



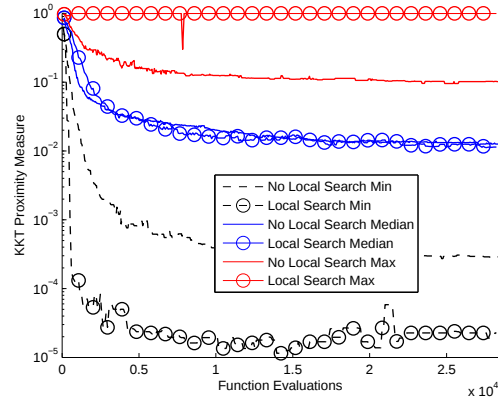
(B) KKTPM for DTLZ2

FIGURE 5.22: Effect of LS on 10-objective DTLZ problems. The hybrid approach shows a better convergence property.

value shows a significant improvement.



(A) KKTPM for CAR (3 obj.)



(B) KKTPM for WELD (2 obj.)

FIGURE 5.23: Effect of LS on two engineering design problems. The hybrid approach shows a better convergence property.

A slight improvement can be seen in the median KKTPM values, while the maximum KKTPM measure results are more or less similar between the two algorithms.

5.4 Summary

In this study, we have proposed a new hybrid EMO procedure that first identifies poorly converged non-dominated solutions and then improves them using an ASF based LS operator. Simulation results on a number of two-objective, three-objective, and 10-objective unconstrained and constrained problems have clearly demonstrated a much faster convergence achieved by our hybrid NSGA-III approach compared to the original NSGA-III approach.

In this study, a recently proposed KKT proximity measure (KKTPM), originally suggested for devising a termination criterion for multi-objective optimization algorithms, has been used to identify the worst-converged non-dominated solution in a population. Such a classification was not possible before and EMO algorithms had to rely on a diversity characterization for differentiating non-dominated solutions. This unique ability facilitated by KKTPM has allowed us to devise an LS based EMO approach to improve the convergence characteristics of the worst-converged population member. The extent of computations has been kept at a low level by allowing only one population member to undergo LS at each generation. It has been observed that for an identical number of solution evaluations, the proposed hybrid NSGA-III with one LS at every generation has been enough to find better-converged population members than the original NSGA-III.

The convergence speed of the hybrid EMO approach as demonstrated here is remarkable and encourages further use of KKTPM in devising improved EMO operators. One disadvantage of the proposed approach is that in some cases, it sacrifices a little on diversity for the sake of convergence. One way to extend the hybrid approach is to introduce

a special handling of diversity preservation as well. As observed through hypervolume plots, the proposed hybrid approach has achieved a better convergence with an initial sacrifice on diversity of non-dominated solutions. Nevertheless, this study has amply shown the importance of KKTPM in identifying poorly converged non-dominated solutions for achieving an increased convergence property of an EMO algorithm. The study should encourage researchers to pay more attention to KKTPM and other theoretical optimality properties of solutions in arriving at better multi-objective optimization algorithms.

In Chapter 6, we conclude our study by proposing Balanced NSGA-III (B-NSGA-III), an algorithm that employs our convergence enhancement procedure (proposed in this chapter) and a novel diversity preservation procedure side by side. B-NSGA-III maintains an adaptive balance between the two procedures according to the problem in hand.

Chapter 6

Multi-Phase Balance of Diversity and Convergence in Multiobjective Optimization

In this chapter, we propose our multi-phased algorithm (B-NSGA-III). In each phase, B-NSGA-III focuses on one multi-objective optimization aspect. With enough temporary evidence that the designated aspect has been fulfilled, the algorithm moves to the next phase dealing with a different aspect. *Phase-1* tries to stretch the non-dominated set of solutions as much as possible by looking for farther extreme points. Once settled, *phases 2 and 3* try to cover the gaps found in the non-dominated front and pull least-converged solutions towards the true Pareto front respectively. The key behind the success of B-NSGA-III is its alternating nature. B-NSGA-III follows the following principle: "At any given point during the course of optimization, the conviction that any of the aforementioned aspects

has been truly fulfilled may *not* be true.". For example, the belief that all extreme points found so far are the true extremes of the Pareto front is only temporary, as better extreme points can be still hiding there awaiting discovery. With this principle in mind, we designed B-NSGA-III so that it can return to any prior phase once it discovers the falsehood of an already established conviction.

B-NSGA-III is available as an Open Source Software for researchers willing to further investigate this path¹.

6.1 Proposed B-NSGA-III

B-NSGA-III follows the general outline of U-NSGA-III. Starting with a randomly generated initial population, B-NSGA-III generates an equal number of offspring individuals (solutions/points) using niche-based tournament selection, Simulated Binary Crossover and polynomial mutation [32]. The two populations are then combined and the ideal point is updated. The combined population goes through non-dominated sorting [2] and the next population is formed by collecting individuals front by front starting at the first front. Since population size is fixed, the algorithm will typically reach a situation where the number of individuals needed to complete the next population is less than the number of individuals available in the front currently being considered. B-NSGA-III collects only as many individuals as it needs using a *niching* procedure. This niching procedure normalizes the objectives of all fronts considered. Then, using a fixed set of evenly distributed

¹<https://www.coin-laboratory.com/evolib>

reference directions (in the normalized objective space) preference is given to those solutions representing the least represented reference directions in the objective space so far. Interested readers are encouraged to consult Chapter 3 for more details.

U-NSGA-III maintains a constant preference of convergence over diversity. A solution in front $n + 1$ will never be considered for inclusion in the next population unless all solutions in front n are already included. This convergence-always-first scheme has been recently criticized by several researchers [3, 106]. Moreover, our approach discussed in Chapter 5 although enhances convergence, it can negatively affect diversity. B-NSGA-III breaks this constant emphasis on convergence. Every α generations, the proposed algorithm changes its survival selection strategy, by favoring solutions solely representing some reference directions over other – possibly dominating – redundant solution. A redundant solution is a solution that is not the best representative of its niche i.e. there exists another solution – in the same front – that is closer to the reference direction representing their niche.

Another difference between U-NSGA-III and B-NSGA-III is that U-NSGA-III treats all individuals/regions of the search space equally at all generations. And although it might seem better from a generic point of view, we claim the opposite. One of the most important resources in optimization is the number of Solution Evaluations (SEs) consumed to reach a solution. Being fair the way U-NSGA-III is, can lead to wasting SEs on easy sections of the Pareto front. Those wasted SEs could have been put into better use, if they were directed towards reaching more difficult sections of the front. Obviously, in order to achieve the maximum possible utilization of SEs, a truly dynamic algorithm that gives more attention to more difficult sections/points of the front is needed. However, designing such an algorithm needs an oracle that knows deterministically the difficulty of attaining

each point on the front relative to others. Unfortunately, for an arbitrary optimization problem, perfecting such an oracle is a far-fetched dream so far, despite some studies [107]. However, recent studies show some clues that can drive creating a non-deterministic version of the targeted oracle. We summarize these clues in the following points:

1. Researchers have repeatedly shown the important role normalization plays especially in achieving better coverage of the Pareto front. Usually, the extreme points of the current population dictate normalization parameters. During evolution, as new extreme points appear, all previously normalized objective values become outdated, and normalization is repeated. Hence, the importance of extreme points in optimization. And as pointed in [108], the earlier we reach extreme points, the better normalization we have and the better coverage we attain.
2. Reaching some parts of the Pareto front may require more effort than others. Several test as well as real world problems exhibit such behavior [99, 104]. Usually such difficult regions appear as gaps in the first front. In reference directions based optimization algorithms (like MOEA/D, NSGA-III and U-NSGA-III), gaps can be identified by looking for reference directions having no associations so far.
3. In multiobjective optimization, all non-dominated solutions are considered equally good, thus deserving equal attention. This is not ideal though. Being non-dominated with respect to each other does not mean that two solutions are equally converged. The recently published approximate KKTPM enables us to efficiently differentiate non-dominated solutions based on their proximity from local optima.

These clues are realized in B-NSGA-III through several phases. In α generations, the algorithm switches back and forth among three different phases. Each phase uses a specific LS operator to fulfill its goal. The following subsection discusses these phases in greater detail.

6.1.1 Alternating Phases

One naive approach is to use sequential phases. Given their relative importance the first phase may seek extreme points. Once found, the algorithm moves to subsequent phases and never looks back. But, as shown in [108], reaching true extreme points is not a trivial task. Even using LS, several optimizations might be needed to attain one extreme point. And since we can never safely assume that we have reached the true extreme points, this sequential design is not recommended. The same argument is valid for covering gaps. In an earlier generation, although your solutions may not provide the desired spread, it might be the case that there are no gaps within the small region they cover. As generations proceed and solutions expand, gaps may appear. This is a frequent pattern that is likely to repeat through an optimization run. Again, the sequential pattern is prone to failure, as we can never know if more gaps will appear in the future.

Another more involved yet simple approach is to move from one phase to the next after a fixed number of generations (or SEs). Once, the algorithm reaches the final phase it goes back to the first cyclically. This cyclic approach is more appealing, but how many generations (or SEs) to wait for before switching from one phase to the next? Obviously, it is never easy to tell. In addition, using this rigid design obligates the algorithm to spend

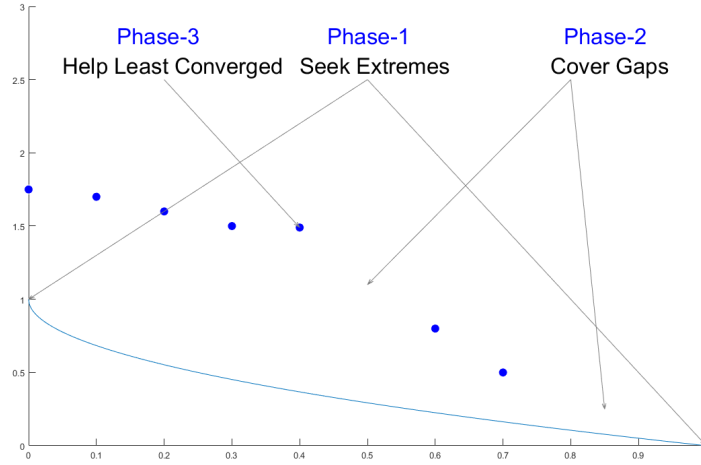


FIGURE 6.1: *Phase-1*, *Phase-2* and *Phase-3* in action

resources (SEs) in *possibly unnecessary* phases, just because it is their turn in the alternation cycle.

B-NSGA-III alternates among three phases *dynamically* and *adaptively*. Figure 6.1 shows the three phases in action. During *Phase-1*, the algorithm seeks extreme points. *Phase-2* is where an attempt is made to cover gaps found in the non-dominated front. Finally, during *Phase-3* the focus is shifted towards helping poorly converged non-dominated solutions. In order to avoid the shortcomings of the two aforementioned approaches, B-NSGA-III watches for specific incidents that trigger transitions from one phase to another. Those transitions are completely unrestricted i.e. B-NSGA-III can move from any phase to the other if the appropriate trigger is observed. Figure 6.2 shows all possible transitions along with their triggers.

In the first α generation, the algorithm puts itself in *Phase-1*. Algorithm 6 shows the details of this phase. For an M objectives problem, *Phase-1* uses an Extreme-LS operator (discussed later) to search for its M extreme points. If all extreme points remain unchanged

ALGORITHM 7 Phases 2 and 3

Input: parents (P), offspring (O), number of objectives (M), population size (N), reference directions (D), ideal point (I), intercepts (T), maximum number of function evaluations ($FeMax$), maximum number of local search operations per iteration β , last point used to cover direction d ($prev_d$) for all d in D

Output: New Population ($\hat{P}$)

```
1:  $F \leftarrow nonDominatedSorting(All)$ 
2:  $All \leftarrow P \cup O$ 
3:  $\hat{P} \leftarrow getBestWithinNiche(d, O) \forall d \in D$ 
4: if  $stagnant(E)$  then
5:    $D_{empty} \leftarrow \{d \in D \mid (\nexists x)[x \in F_1 \wedge x \in d_{surroundings}]\}$ 
6:    $s_{kktpm} \leftarrow calculateKKTpm(s) \quad \forall s \in \hat{P}$ 
7:   for  $i = 1$  to  $\beta$  do
8:     if  $D_{empty} \neq \phi$  then ► Phase-2
9:        $d \leftarrow randomPick(D_{empty})$ 
10:       $s \leftarrow \{x \in F_1 \mid$ 
11:         $(\nexists y)[y \in F_1 \wedge \perp_d(y) < \perp_d(x)]\}$ 
12:      if  $prev_d = null$  or  $\perp_d(s) < \perp_d(prev_d)$  then
13:         $prev_d \leftarrow s$ 
14:      else
15:         $s \leftarrow null$ 
16:         $D_{empty} \leftarrow D_{empty} \setminus \{d\}$ 
17:      end if
18:    else ► Phase-3
19:       $s \leftarrow \{x \in \hat{P} \mid$ 
20:         $(\nexists y)[y \in \hat{P} \wedge y_{kktpm} > x_{kktpm}]\}$ 
21:      end if
22:      if  $s \neq null$  then
23:         $\hat{s} \leftarrow localSearch_{ASF}(s, I, T, FeMax)$ 
24:         $\hat{P} \leftarrow \hat{P} \cup \{\hat{s}\}$ 
25:         $\beta \leftarrow \beta + 1$ 
26:      end if
27:    end for
28:  end if
```

solely represent their niche, every α generations. This is shown in Algorithm 7, line 3 and expanded in Algorithm 8. The points surrounding each *non-empty* reference direction are collected from the merged population (parents and offspring) (line 2), and the best ranked point is selected to represent this direction/niche (line 4). If more than one point share the same rank, the point closest to the direction is selected (line 5). Obviously, as opposed to U-NSGA-III points in B-NSGA-III compete only with their niche peers, which means

that an inferiorly ranked point from one niche, can be included because it is the best representative of its niche, while a superior point from another niche is left out because a better representative of its niche exists.

Once in *Phase-2*, B-NSGA-III looks for reference directions having no associations in the first front (*empty directions*). These directions represent gaps in the non-dominated front (Algorithm 7, line 5). If several such directions exist, one is picked randomly (line 9) and the closest first front point to this direction is saved (line 10) to be used later as a starting point in LS. Notice that lines 11 to 16 ensures that B-NSGA-III will not re-try to cover a gap until a closer starting point than the one previously used is found. If no empty directions exist, B-NSGA-III moves to *Phase-3*, looking for the least converged point among those selected so far. This point should have the highest KKTPM among all (line 18). An ASF-LS operator (discussed later) is employed in both cases (line 21), either to cover a gap (*Phase-2*) or to bring a poorly converged point closer to the front (*Phase-3*). In order to keep SEs as low as possible, a maximum of β LS operations are allowed, even if the number of gaps is more than β . Notice the ability of the algorithm to move directly from *Phase-1* to *Phase-3* if no gaps are found.

ALGORITHM 8 getBestWithinNiche(All, D)

Input: merged parents and offspring (All), reference directions (D)

Output: selected Individuals (one from each niche) ($\hat{P}$)

```

1:  $\hat{P} \leftarrow \phi$ 
2:  $S \leftarrow getSurroundings(d, All)$ 
3: for all  $d \in D$  do
4:    $X_d \leftarrow \{x \in S \mid (\nexists y)[y \in S \wedge y_{rank} < x_{rank}]\}$ 
5:    $x_d \leftarrow \{x \in X_d \mid (\nexists y)[y \in S \wedge \perp_d(y) < \perp_d(x)]\}$ 
6:    $\hat{P} \leftarrow \hat{P} \cup \{x_d\}$ 
7: end for
```

It is worth noting that phases 2 and 3 may run simultaneously. If the number of empty

ALGORITHM 9 fillUpPop(All , N , $\hat{P}$)

Input: merged parents and offspring (All), population size (N), partially full new population ($\hat{P}$)

Output: completely full new population ($\hat{P}$)

```
1:  $All \leftarrow All \setminus \hat{P}$ 
2: while  $|\hat{P}| < N$  do
3:    $Z \leftarrow \{x \in All \mid (\nexists y)[y \in All \wedge y_{rank} < x_{rank}]\}$ 
4:    $z \leftarrow pickRandom(Z)$ 
5:    $All \leftarrow All \setminus \{z\}$ 
6:    $\hat{P} \leftarrow \hat{P} \cup \{z\}$ 
7: end while
```

directions (gaps) is less than β , B-NSGA-III moves to *Phase-3* and uses the remaining budget to help poorly converged solutions. Obviously, *Phase-1* has the highest priority followed by *Phase-2* then *Phase-3*. The following two subsections discuss both Extreme-LS and ASF-LS operators in detail.

Finally, since all the three phases are not guaranteed to completely fill the next population, a final pass is made to fill up the next population using points that B-NSGA-III has discarded so far. Algorithm 9 shows that the best ranked points – out of those not included yet in the next population – are given higher priority.

6.1.2 Two Local Search Operators

As mentioned earlier, B-NSGA-III uses two different local search operators. In each, all the objectives are combined into some aggregate function (scalarization). Any single objective optimizer can be used to minimize these aggregate functions. Here we chose to use the same Matlab's[®] `fmincon()` optimization routine used in Chapter 5, a point-to-point deterministic optimizer. Point-to-point optimizers use less function evaluations compared to set-based methods (e.g. evolutionary algorithms). But, they are also less guaranteed to reach global optima. Yet, in an alternating multi-phased algorithm like

B-NSGA-III the embedded single objective optimizer is not expected to reach the global optimum in one shot. That's why `fmincon()` fits our criteria for an embedded single objective optimizer. Earlier we discussed the role of our two LS operators. Next we discuss their formulations and how they fit into their designated roles.

6.1.2.1 Extreme-LS

Phase-1 uses Extreme-LS to find extreme points. This operator is formulated simply as a Biased Weighted Sum (BWS) aggregate function of all objectives (Equation 6.1). $\tilde{f}_k(x)$ represents the normalized value of objective k . When seeking the i^{th} extreme point, the term *Biased* refers to the significantly smaller weight (we call it *augmentation factor*) multiplied by the i_{th} objective, compared to the weights of all other objectives. Although, weighted sum aggregate functions are straightforward and easy to implement, they (including ours) can only reach points lying on convex sections of the Pareto front. While, this makes them less plausible as a generic formulation, they perfectly serve their purpose in B-NSGA-III, since extreme points by definition can never lie in a non-convex section of the Pareto front. Unlike [109], the operator we are proposing here is normalized based on the number of objectives. This allows using the same *augmentation factor* for different dimensions. It is important to note that adding the i -th objective term to the formula helps avoiding weakly dominated points.

$$\underset{\mathbf{x}}{\text{Minimize}} \quad \text{BWS}_i(\mathbf{x}) = \epsilon \tilde{f}_i(x) + \sum_{j=1, j \neq i}^M \frac{w_j \tilde{f}_j(x)}{M-1}, \quad (6.1)$$

where ϵ is set as one percent of $\min_{j=1, j \neq i}^M w_j$.

6.1.2.2 Achievement Scalarization Function LS (ASF-LS)

As mentioned earlier, a generic LS operator that is required to get an arbitrary Pareto point cannot rely on BWS. That's why we use ASF to formulate our second LS operator, ASF-LS. The formulation in Equation 6.2 shows that ASF-LS targets the intersection between the provided direction and the Pareto front. Since, B-NSGA-III is a reference direction based algorithm, ASF-LS can follow these already existing directions. And because of its ability to reach points lying on both convex and non-convex sections of the Pareto front, this is the operator employed in both *Phase-2* and *Phase-3* of B-NSGA-III. It is worth noting that according to our earlier experiments, ASF-LS does not perform as well if used to find extreme points. This can be attributed to the steep gradient of the aggregate ASF function (at these points) on one side of the global optimum, which usually misleads the single objective optimizer. Hence, we need both operators in B-NSGA-III, each playing its designated role.

$$\begin{aligned} \text{Minimize}_{\mathbf{x}} \quad & \text{ASF}(\mathbf{x}, \mathbf{z}^r, \mathbf{w}) = \max_{i=1}^M \left(\frac{\tilde{f}_i(\mathbf{x}) - u_i}{w_i} \right), \\ \text{subject to} \quad & g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, J. \end{aligned} \tag{6.2}$$

6.2 Results

The set of problems used here are carefully selected and/or modified to exhibit different types of difficulties. Both unconstrained and constrained problems are taken into account. Some of them have disconnected Pareto fronts. Others, challenge the convergence ability of the multiobjective algorithm, while others test its ability to cover the entire front evenly

(diversity). Some of these problems are differentiable while others are not. We have even modified some problems to exhibit certain types of behavior, different from what they were originally designed for. Our problems cover a wide range of dimensionality as well, namely 2, 3, 5 and 10 objectives. B-NSGA-III is compared to two other state-of-the-art reference direction based algorithms, U-NSGA-III and MOEA/D. Since the original study [3] did not mention a specific way to handle constraints, we use MOEA/D with unconstrained problems only. U-NSGA-III on the other hand is applied to all problems. Population size is kept at a small value to maintain a strict testing conditions on all algorithms. We use Inverted Generational Distance (IGD) and Generational Distance (GD) to test both overall performance and convergence respectively. Sometimes, we also show how KKTPM progresses during optimization. All results presented here are the medians of 31 independent runs of each algorithm on each problem. Table 6.1 shows the parameters used by B-NSGA-III in each problem. The other two algorithms use the same values for common parameters.

Before going through the details of our simulations, we emphasize the alternating nature of B-NSGA-III by showing it in action. Figure 6.3 shows a typical alternation of phases performed by B-NSGA-III while solving ZDT4 [104].

6.2.1 Multi-objective problems

We start our experiments by solving three unconstrained bi-objective problems, ZDT3, ZDT4 and ZDT6 [99] to test the ability of our algorithm to deal with disconnected Pareto fronts, local Pareto fronts and variable density objective spaces respectively. As shown in

²Using numerical derivatives.

TABLE 6.1: Parameters used by B-NSGA-III. M is the number of objectives. N is population size. SEs stands for Solution Evaluations. α is the frequency explained in Section 6.1 while β is the maximum limit of function evaluations the single objective optimizer (`fmincon()`) can use. The final column shows the values of ϵ in Equation 6.1.

Problem	M	N	SEs	α	$FE_{\max}$	ϵ
ZDT3	2	72	8000	10	200	0.1
ZDT4	2	48	30000	10	200	0.1
ZDT6	2	48	5000	10	200	0.1
OSY	2	48	15000	10	200	0.1
TNK	2	24	5000	10	200	0.1
DTLZ4	3	36	7000	10	200	0.1
DTLZ4	5	120	25000	10	200	0.1
DTLZ4	10	276	50000	10	200	0.1
DTLZ7	3	92	20000	10	200	0.1
WFG1	3	92	20000	10	200	0.1

TABLE 6.2: Each problem test one or more aspects of the algorithm

Problem	Properties
ZDT3	disconnected Pareto Front (PF)
ZDT4	large number of local PFs, distant initial population
ZDT6	biased density objective space
OSY	PF sections vary in difficulty
TNK	disconnected PF
DTLZ4	biased density objective space, distant initial population
DTLZ7	disconnected PF, PF sections vary in difficulty
WFG1	non-differentiable, local PFs, differently scaled objectives

TABLE 6.3: p-values of a two sided Wilcoxon rank sum test.

Problem	M	B-NSGA-III vs. U-NSGA-III	B-NSGA-III vs. MOEA/D
ZDT3	2	3.98×10^{-1}	6.47×10^{-11}
ZDT4	2	1.40×10^{-11}	1.68×10^{-9}
ZDT6	2	1.40×10^{-11}	1.99×10^{-5}
OSY	2	1.51×10^{-7}	—
OSY ²	2	7.57×10^{-6}	—
TNK	2	4.32×10^{-4}	—
DTLZ4	3	2.40×10^{-5}	1.51×10^{-7}
DTLZ4	5	1.40×10^{-11}	1.40×10^{-11}
DTLZ4	10	6.47×10^{-11}	1.40×10^{-11}
DTLZ7	3	1.11×10^{-7}	1.08×10^{-4}
WFG1	3	5.38×10^{-2}	1.40×10^{-11}

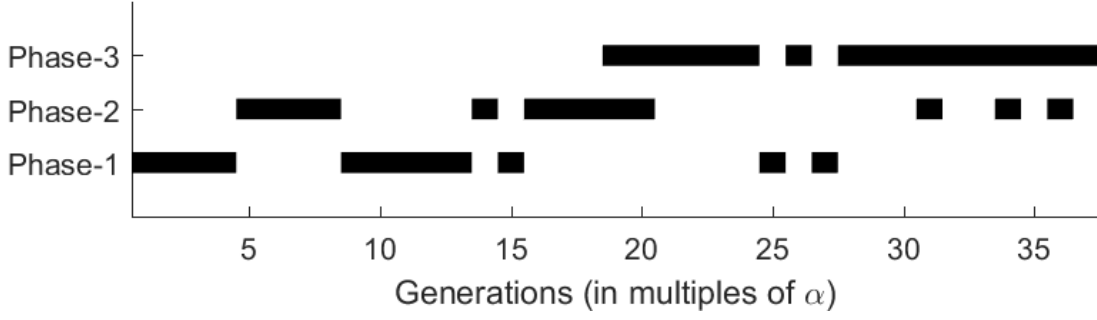


FIGURE 6.3: A typical alternation of phases of B-NSGA-III while solving ZDT4

Figure 6.4, in ZDT3, B-NSGA-III and U-NSGA-III achieve better convergence compared to MOEA/D. The slight difference observed in IGD is not statistically significant (see Table 6.3). In both ZDT4 and ZDT6, B-NSGA-III is a clear winner in terms of both convergence and diversity as shown in Figures 6.5 and 6.6.

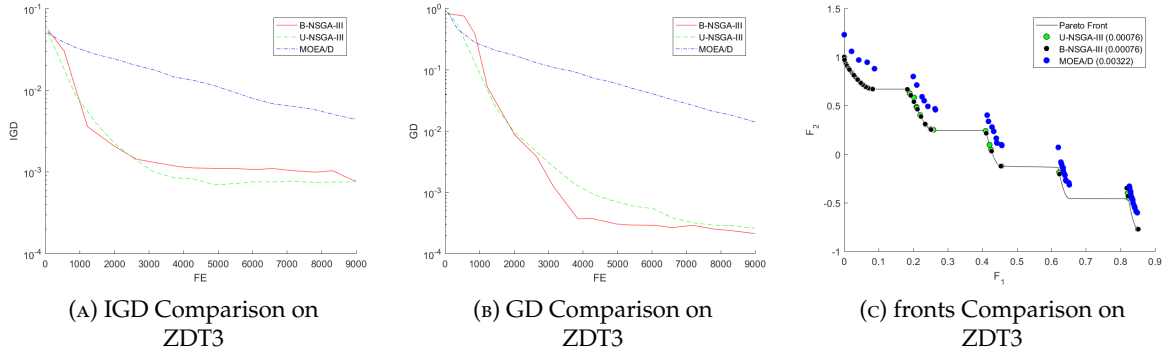


FIGURE 6.4: Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on bi-objective ZDT3

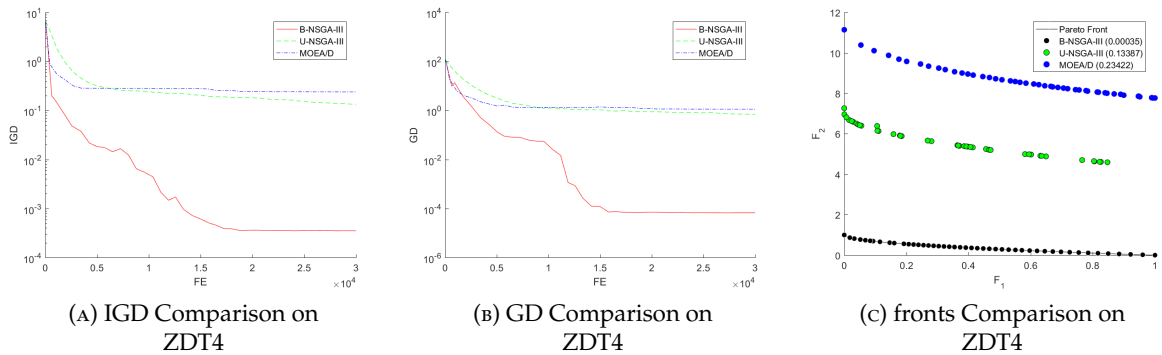


FIGURE 6.5: Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on bi-objective ZDT4

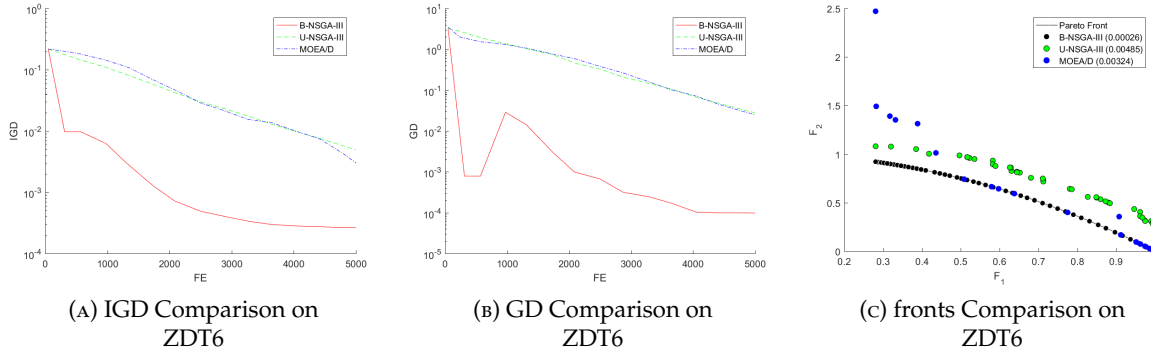


FIGURE 6.6: Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on bi-objective ZDT6

The same outcome can be observed with constrained problems, see Figures 6.7 and 6.8. For these two problems, TNK and OSY [5], we exclude MOEA/D since the original study did not provide a specific way to handle constraints. Notice that for simple problems requiring no special attention to either convergence or diversity, B-NSGA-III and U-NSGA-III behave similarly, which is expected. However as problems get more difficult the merits of B-NSGA-III becomes evident. Notice, how B-NSGA-III was able to cover the entire Pareto front providing a nice distribution in Figures 6.5c, 6.6c and 6.8c using a limited number of function evaluations (see Table 6.1). Both, U-NSGA-III and MOEA/D need more solution evaluations to catch up with B-NSGA-III, if they ever do.

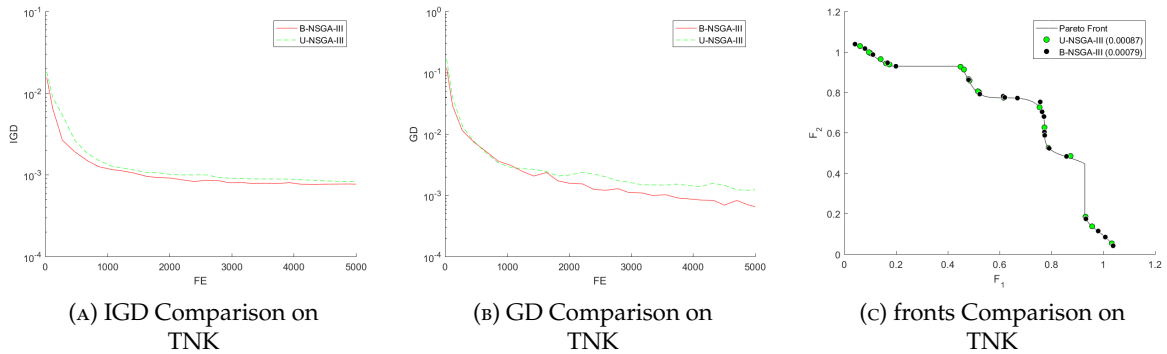


FIGURE 6.7: Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on bi-objective TNK

Looking at KKTPM yields another perspective that can confirm the convergence edge

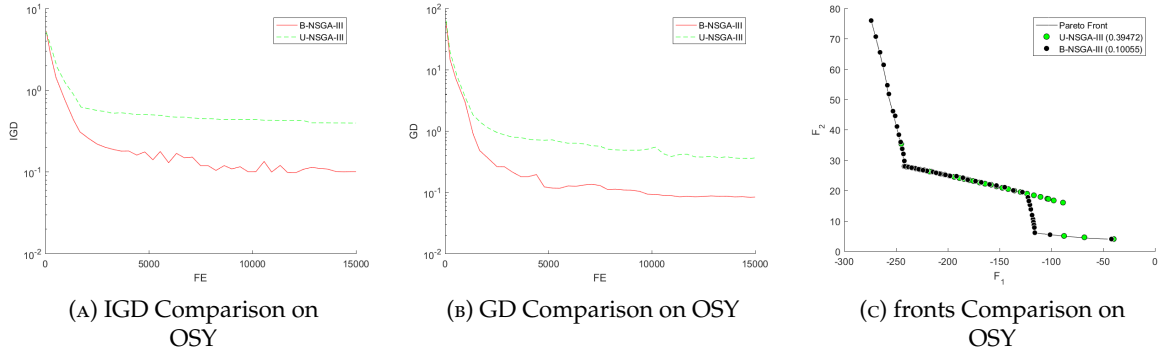


FIGURE 6.8: Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on bi-objective OSY

B-NSGA-III has. Figures 6.9a, 6.9b and 6.9c show how median KKTPM of non-dominated solutions progresses throughout optimization in ZDT4, ZDT6 and OSY, respectively. Obviously, B-NSGA-III can reach Pareto optimal points (either local or global) much faster than U-NSGA-III and MOEA/D. Combined with the previous GD plots, the reader can easily conclude that B-NSGA-III hits global Pareto optimal points much earlier than U-NSGA-III and MOEA/D. Fluctuations can be seen in initial generations (see Figure 6.9b) because of the rapidly changing limited count of non-dominated solutions at early generations.

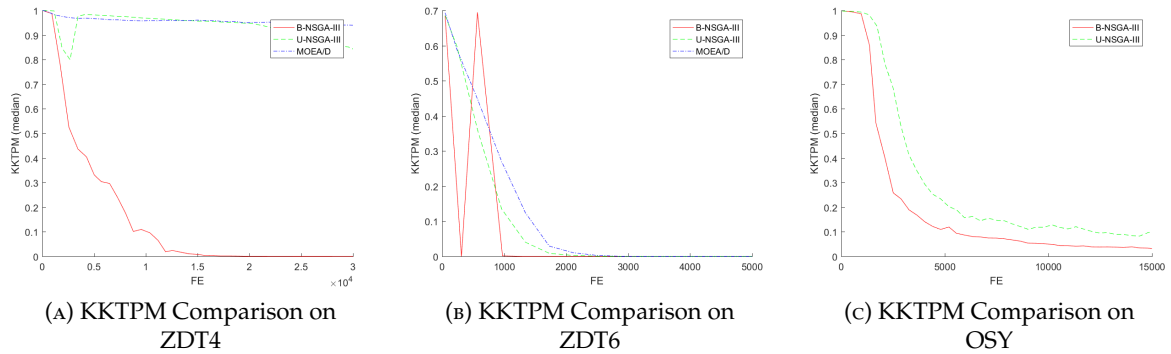


FIGURE 6.9: How median KKTPM progresses over generations in ZDT4, ZDT6 and OSY

For three objectives, we use DTLZ4 and DTLZ7 problems [104]. DTLZ4 tests the ability of an optimization algorithm to diversify solutions in a variable density objective space.

A randomly generated set of Pareto optimal solutions will be dense near the f_M - f_1 plane, and as you move away from it, solutions get sparser. Degree of density/sparseness can be controlled via the parameter γ (called α in the original study and changed here not to be confused with our α parameter shown in Table 6.1). However, DTLZ4 does not test the ability of an algorithm to converge. Actually, all randomly generated solutions are relatively close to the Pareto front (compared to other problems from the same family, like DTLZ1). And since B-NSGA-III is designed to tackle both convergence and diversity simultaneously, we need a problem that is able to test both capabilities at the same time. We can get such a problem by multiplying the $g(x)$ function of DTLZ4 by a constant factor D . The larger D is, the more distant randomly generated solutions will be from the Pareto front. In this study we use $\gamma = 20$ and $D = 100$. On the other hand, the Pareto front of DTLZ7 has four disconnected regions. One region is relatively easier than the others. An optimization algorithm can easily get attracted to the easily attainable region and ignore some/all others.

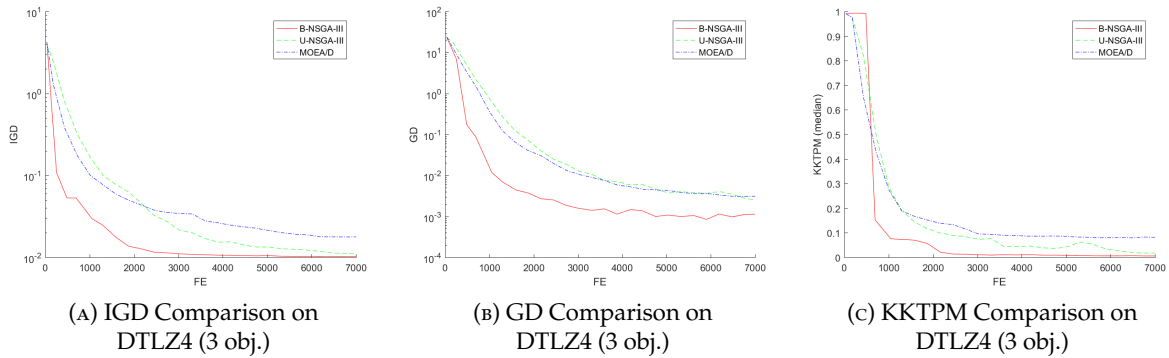


FIGURE 6.10: Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on DTLZ4 (3 objectives).

Figures 6.10a, 6.10b and 6.10c shows the overall performance of B-NSGA-III compared to U-NSGA-III and MOEA/D on DTLZ4. B-NSGA-III is better in terms of both overall performance and convergence. Median Pareto fronts (those having median IGD values)

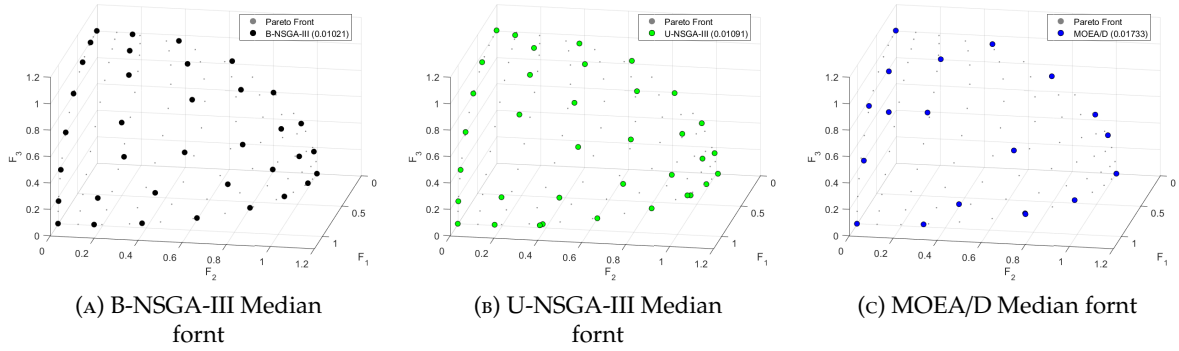


FIGURE 6.11: Median final fronts of U-NSGA-III, and MOEA/D on DTLZ4 (3 objectives).

shown in Figures 6.11a, 6.11b and 6.11c confirm the overall superiority of B-NSGA-III and U-NSGA-III over MOEA/D, especially in terms of diversity. The superiority of B-NSGA-III is even more evident on DTLZ7 as show in Figures 6.12 and 6.13.

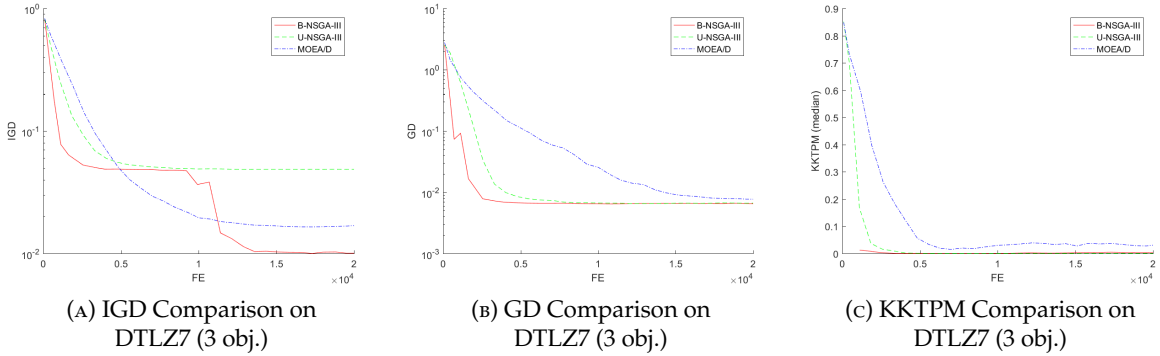


FIGURE 6.12: Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on DTLZ7 (3 objectives).

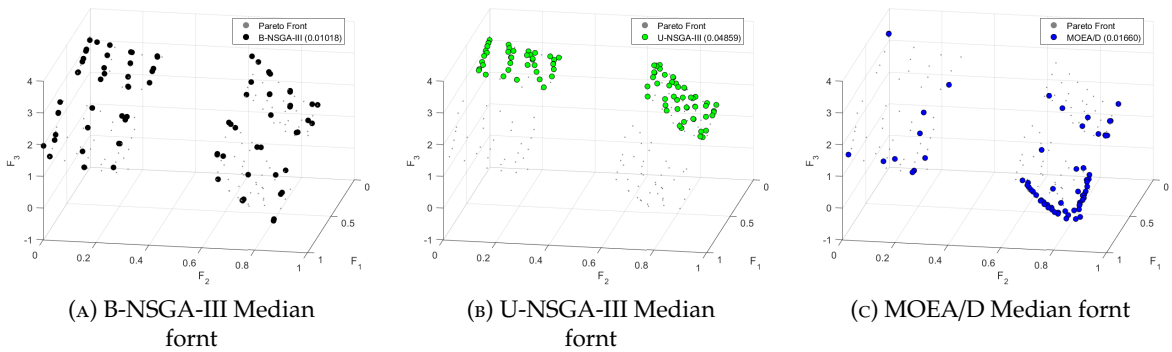


FIGURE 6.13: Median final fronts of U-NSGA-III, and MOEA/D on DTLZ7 (3 objectives).

6.2.2 Many-Objective Optimization

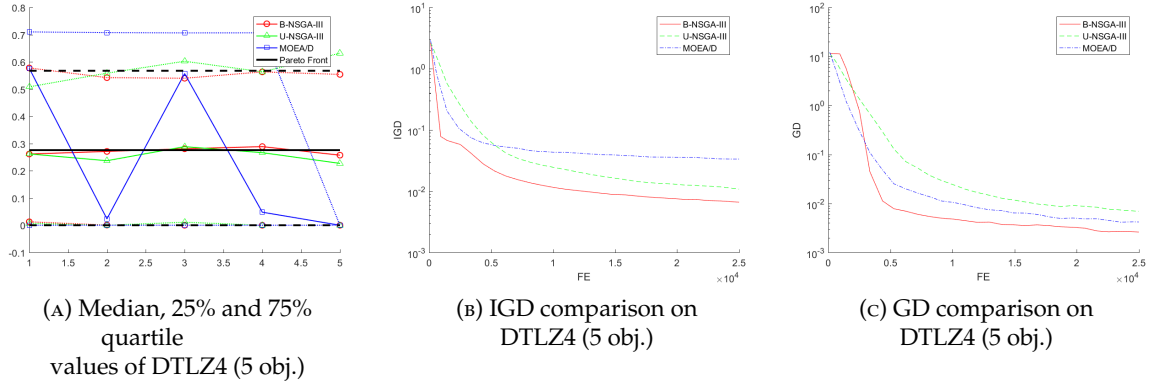


FIGURE 6.14: Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on DTLZ4 (5 objectives).

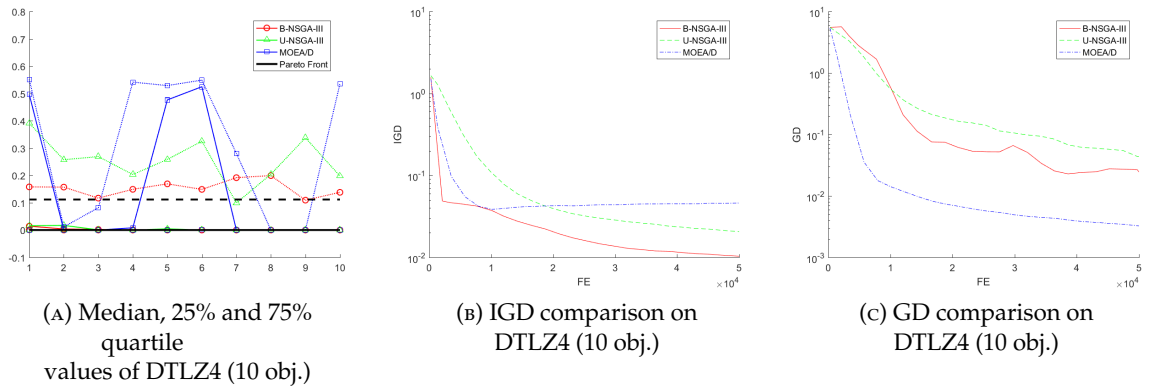


FIGURE 6.15: Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on DTLZ4 (10 objectives).

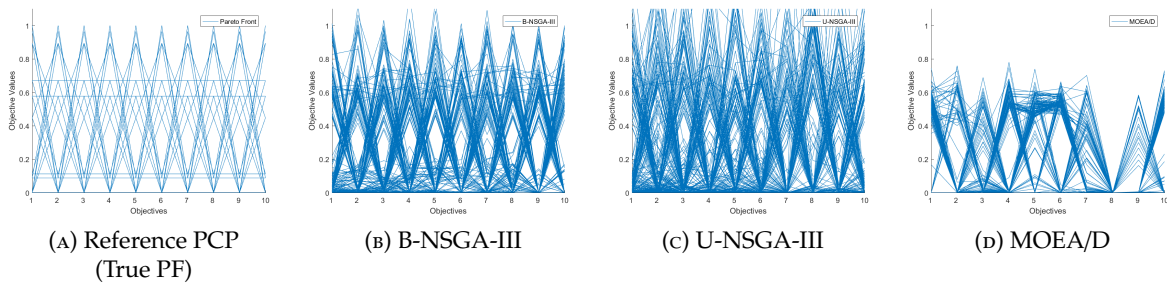


FIGURE 6.16: PCPs of B-NSGA-III, U-NSGA-III, and MOEA/D on DTLZ4 (10 objectives).

A many-objective optimization problem is a problem having more than 3 objectives. Both MOEA/D and U-NSGA-III are known to be very efficient with this category of problems. In

this section, we consider a difficult problem (our modified version of DTLZ4) and compare the performance of B-NSGA-III to these two powerful algorithms. Here, we use two instances of the problem, one with 5 objectives and the other with 10. For the 5 objectives instance, Figure 6.14a is a Parallel Coordinate Plot (PCP) showing the 25%, median and 75% quantile values of each objective among the three algorithms. The values plotted for each algorithm are taken from the final population of the run having the median IGD value of its 31 independent runs. Unmarked lines represent the benchmark quantile values of the selected Pareto optimal set. Lines marked with circles, squares and triangles represent B-NSGA-III, U-NSGA-III and MOEA/D respectively. The closer the lines representing one algorithm to the benchmark lines, the better this algorithm performs. All three algorithms successfully attain the 25% quantile values. However median and 75% quantile results vary. Obviously, B-NSGA-III achieves the best approximation of the two benchmark lines among the three algorithms. U-NSGA-III is still acceptable while MOEA/D misses the target lines completely. In addition, IGD and GD median comparisons are shown in Figures 6.14b and 6.14c respectively. Obviously, B-NSGA-III outperforms the other two algorithms in terms of overall performance and convergence.

The same conclusions can be drawn for the 10 objectives problem instance. Since most of the Pareto points are on the edges of the hyper simplex, the median of each objective is the same as the minimum (Zero). Figure 6.15a shows that unlike both U-NSGA-III and MOEA/D, B-NSGA-III achieves a close approximation of the benchmark lines. Those results are supported by IGD and GD results in Figures 6.15b and 6.15c respectively. In order to further confirm our results, we have included the PCPs of all non-dominated solutions of the median run of each algorithm in Figure 6.16. Obviously, the PCP of

B-NSGA-III in Figure 6.16b is closer to the benchmark PCP (Figure 6.16a) than both the PCPs of U-NSGA-III and MOEA/D.

6.2.3 Using numerical gradients

Up to this point, our approach has been using exact analytical gradients – provided by the user – to calculate KKTPM. However for many problems – especially real-world problems – these gradients may not be available in their analytical form. In such cases we resort to using numerical gradients. In most cases numerical gradients yield acceptable gradient approximations, but this comes on the expense of consuming more SEs. In order to investigate the effect of using numerical gradients on our approach, first we use numerical gradients with a problem for which we know the exact analytical gradients, namely OSY. Then we apply the same approach to WFG1 [110], a problem with non-differentiable objective functions.

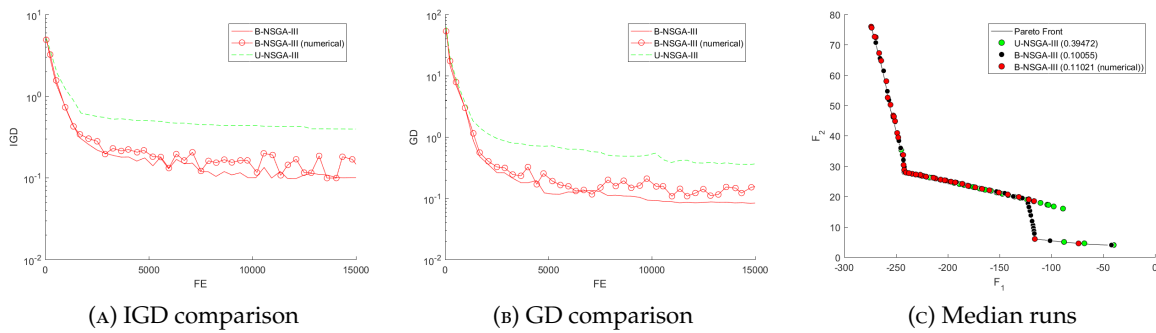


FIGURE 6.17: Performance of B-NSGA-III (exact & numerical) and U-NSGA-III on OSY problem.

Figure 6.17 clearly show how using numerical gradients has a minor negative effect on the performance of B-NSGA-III. As the reader can observe, B-NSGA-III using numerical gradients still outperforms U-NSGA-III. In addition, although using numerical gradients consumes more SEs, it is clear from both figures that the magnitude of sacrifice is minimal.

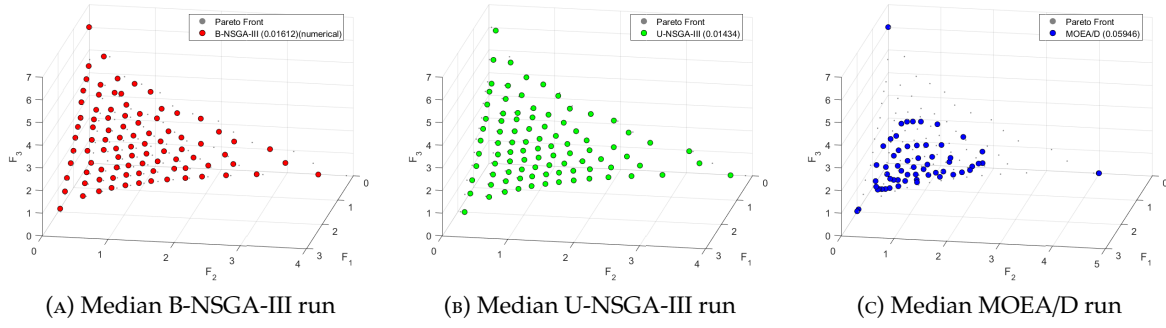


FIGURE 6.18: Performance of B-NSGA-III (using numerical gradients), U-NSGA-III, and MOEA/D on WFG1 problem.

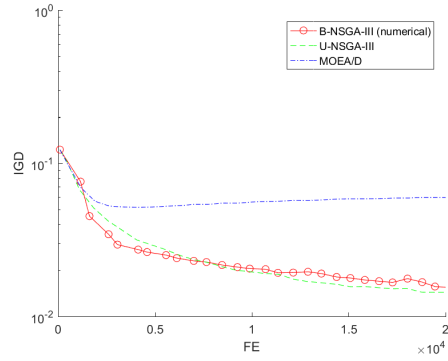


FIGURE 6.19: IGD of the three algorithms on WFG1

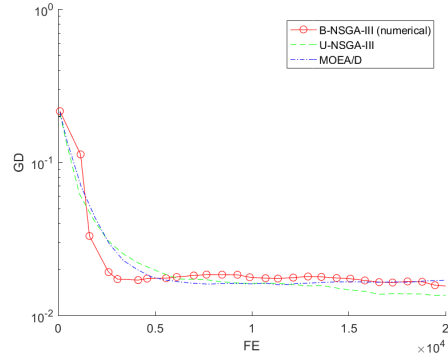


FIGURE 6.20: GD of the three algorithms on WFG1

B-NSGA-III with numerical gradients even catches up with the exact gradients version in some cases.

The reader can easily reach a similar conclusion by looking at Figures 6.18, 6.19 and 6.20. Here we use a slightly modified version of WFG1. We noticed that in order to reach about

100 Pareto Front points in the original version of WFG1, we can only use a code whose precision goes beyond $\approx 10^{-50}$! This value represents the difference between objective values of two neighboring points in such a front. This is true because of the very small power used at the third transformation of WFG1 (0.02). To the best of our knowledge, most of the codes available are not even close to this level of precision. In order to solve this problem we changed the power of the third transformation from 0.02 to 0.2.

Since, all the objectives of WFG1 are non-differentiable, the only possible option for B-NSGA-III is to use numerical gradients. Even with the burden of additional solution evaluations, B-NSGA-III (with numerical gradients) still outperforms both MOEA/D and U-NSGA-III. It is worth noting that, although Figures 6.18a and 6.18b look similar, the points obtained by B-NSGA-III are more converged (closer to the front) than those obtained by U-NSGA-III.

The good performance of numerical gradient based B-NSGA-III can be attributed to the very conservative approach B-NSGA-III follows with KKTPM calculations, which is summarized in the following points:

- Since our modified niching approach is applied every α generations, KKTPM calculations are only possible in $1/\alpha$ of the total number of generations. Typically, in our case – since $\alpha = 10$ – KKTPM values may be calculated in only 10% of all generations at max.
- Even when it comes to these 10% of generations, *Phases-1* and 2 do not require any KKTPM calculations. And as we showed earlier, in most problems B-NSGA-III spends the majority of its time in these two phases.

- *Phase-3* is where most KKTPM calculations are required. In this phase, KKTPM values are calculated only for those individuals selected so far, which are usually less than N (population size).

6.3 Summary

This chapter concludes our contributions by proposing B-NSGA-III, a multi-phased many-objective evolutionary optimization algorithm capable of automatically balancing convergence and diversity of population members. B-NSGA-III does not use explicit preferential parameters or weights to distribute its effort among the two aspects. It rather waits for signals based on which it changes from one phase to the other. This alternation of phases is completely unrestricted and the exact way it happens adapts automatically to the problem in hand. Two types of local search operators are used during optimization. The first is designed to find extreme points while the second is more suited to move arbitrary points towards the Pareto optimal front. Approximate KKTPM is used to identify weakly dominated points that need to be locally enhanced. On a wide range of problems with different attributes and difficulties, B-NSGA-III shows superior performance to a number of state-of-the-art algorithms for an equal number of solution evaluations.

The specific algorithmic setup proposed in this study should be viewed as one among many possibilities by which several optimization techniques, metrics and algorithms can communicate and cooperate to reach better results. We argue that our open source implementation of B-NSGA-III can serve as a starting point for researchers willing to further exploit these possibilities towards more balanced many-objective optimization.

Chapter 7

Conclusions and Future Work

Throughout the chapters of this study, we were able to devise a unified evolutionary optimization algorithm that can efficiently scale up from one to many objectives (Chapter 3). We have also provided detailed evidence on the role selection plays in our algorithm (U-NSGA-III) and its predecessor (NSGA-III), while further justifying our approach towards a successful unified algorithm (Chapter 4).

Then we shifted our focus to the ability of our proposed approach to balance convergence and diversity in multi- and many-objective optimization problems. First, we proposed a novel convergence enhancement mechanism using a combination of the recently proposed KKTPM and a local search operator using a point-to-point optimizer (Chapter 5). Finally, we extended our approach by employing an additional diversity preservation procedure that works along side the convergence enhancement mechanism. The final product of our study is B-NSGA-III, a unified scalable multi-phase optimization algorithm that can

distribute emphasis between convergence and diversity automatically based on the problem (Chapter 6). B-NSGA-III provides seamless transitions from one phase to another maintaining a high degree of flexibility that suites a wide range of problems regardless of their difficulty.

All our hypothesis and algorithms were logically developed then tested using extensive set of simulations on a wide range of single, multi- and many-objective optimization problems. Both constrained, unconstrained, differentiable and non-differentiable problems were included.

Finally, in order to boost this field of research, we provide open source implementations of all the ideas (algorithms) proposed in this study ^{1 2 3 4}. Interested researchers aided with these open source libraries, can explore unprecedented combinations of evolutionary operators, local search mechanisms and optimality metrics among others. These combinations can open new horizons of single, multi- and many-objective optimization.

7.1 Future Work

In the course of our work on different parts of this study, some interesting questions arose. Will it be beneficial to study diversity in the genotypic (design) space, as opposed to the focus of our study on phenotypic diversity? This approach can be useful with certain categories of problems. For example, in design problems, this approach can reveal

¹EvoLib (coin-laboratory.com/evolib): An open source implementation of NSGA-III, U-NSGA-III and B-NSGA-III

²KKTPM Calculator (coin-laboratory.com/kktpm): An open source implementation of Karush Kuhn Tucker Proximity Measure

³Tx2Ex (github.com/000haitham000/tx2ex): An open source library for parsing and evaluating complex mathematical expressions efficiently and on the fly.

⁴CMA-ES and MOEA/D runs were conducted using JMetal (jmetal.github.io/jMetal) and MOEA Framework (moeaframework.org) libraries respectively.

innovative designs. Being innovative means that these solutions lie on unattended regions of the design (genotypic) space. These designs can be overlooked if the usual objective (phenotypic) space diversity is used, just because they fall in the same vicinity of other non-dominated solutions.

Another question that needs more investigation in reference-based EMO is how to distribute an arbitrary number of reference directions uniformly on the Pareto front. The currently used Das and Dennis approach allows only for a specific number of points depending on the objective dimensionality of the problem. This situation becomes more complicated with discontinuous Pareto fronts.

A very promising line of research is to devise a method to re-allocate reference directions if their original locations are not as fruitful as expected e.g. assigned to gaps where not Pareto points exist. Although the original NSGA-III study provided some insights in this regard they remain limited, leaving significant potential of finding more innovative approaches.

A fourth direction is to mitigate the rapid increase in solution evaluations as objective dimensionality grows. This can be done through combining many-objective optimization with meta-modeling methods. A carefully crafted combination can enable us to solve many objective problems with much lower budget than what we currently use.

Finally, given the contributions of our study, interested researchers must be urged to extend the ideas and concepts used here to massive objectives (25+ objectives). This extension is not expected to be easy. The gap we need to cover to move from many to

massive objectives might be even bigger than the gap that used to exist between multi and many objectives.

All these questions and more need further research and investigation. Hence, they are promising extensions to this study.

APPENDIX

Appendix

Additional figures and tables of Chapter 3

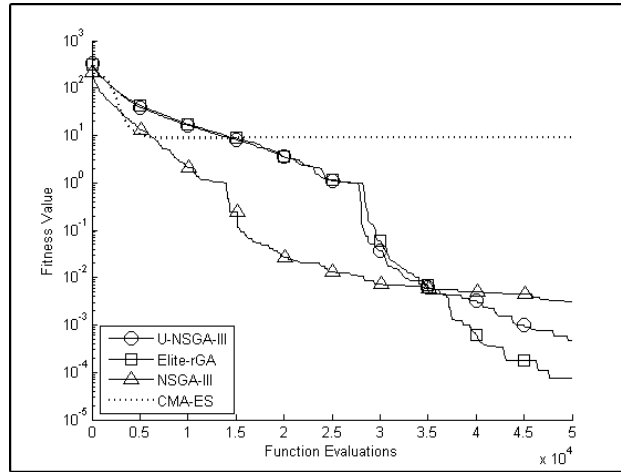


FIGURE A.1: Performance comparison between EliteRGA, NSGA-III, U-NSGA-III and CMA-ES on Rastrigin's unconstrained multimodal problem

TABLE A.1: Single objective unconstrained significance test (p – value)

Problem	U-NSGA-III vs. NSGA-III	U-NSGA-III vs. EliteRGA	U-NSGA-III vs. CMA-ES
ELL	0.5957	0.2073	8.0606E-10
ROS	0.6625	0.2975	5.0518E-13
ZAK	3.2751E-5	0.3528	5.0402E-13
SCH	5.0518E-13	0.0816	6.0875E-15
RAS	9.2668E-5	0.3267	5.7257E-8

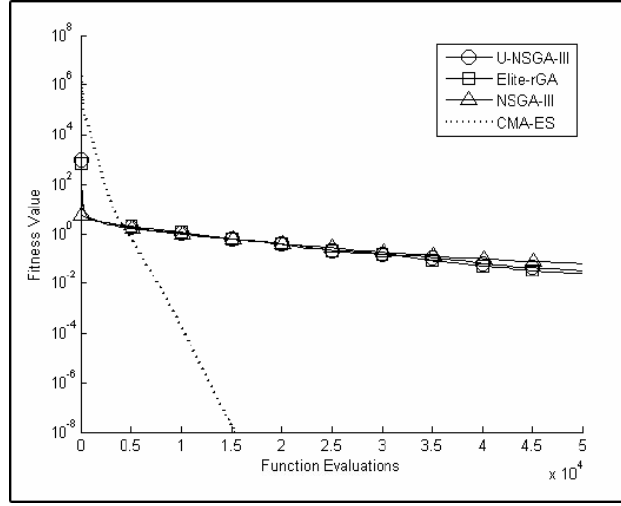


FIGURE A.2: Performance of EliteRGA, NSGA-III, U-NSGA-III and CMA-ES on Zakharov's unconstrained test problem

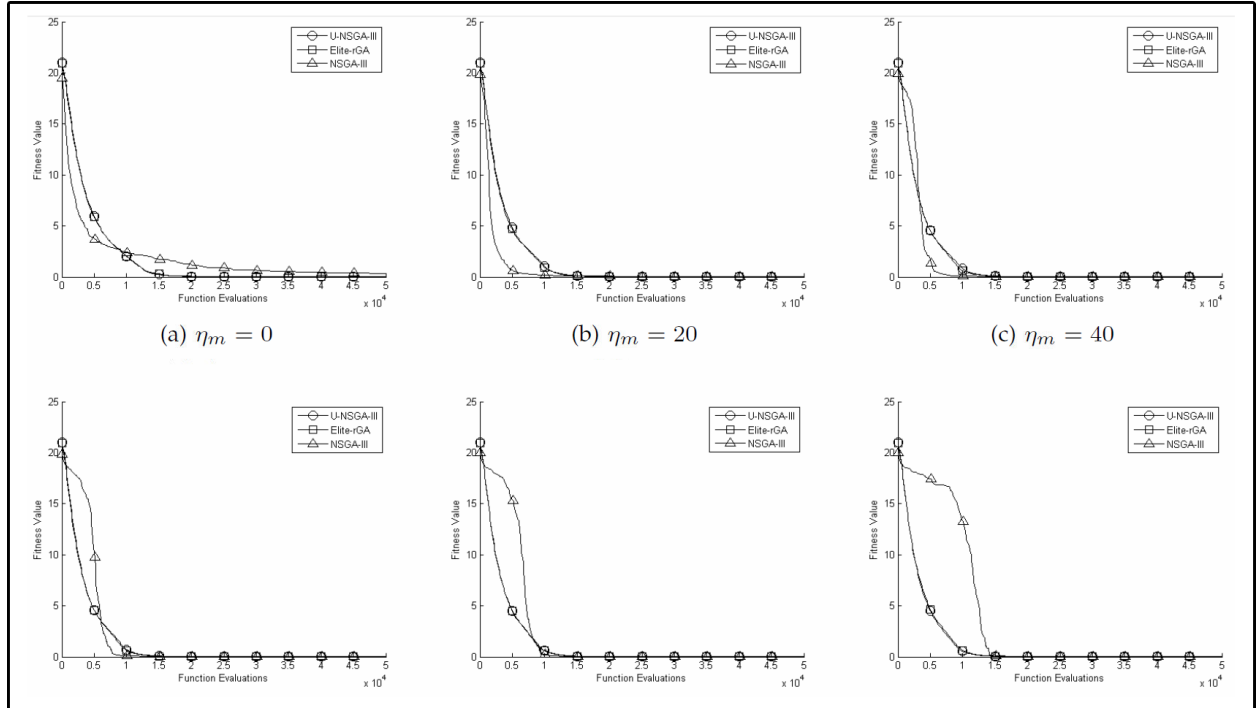


FIGURE A.3: Effect of increasing polynomial mutation distribution index on Ackley's function

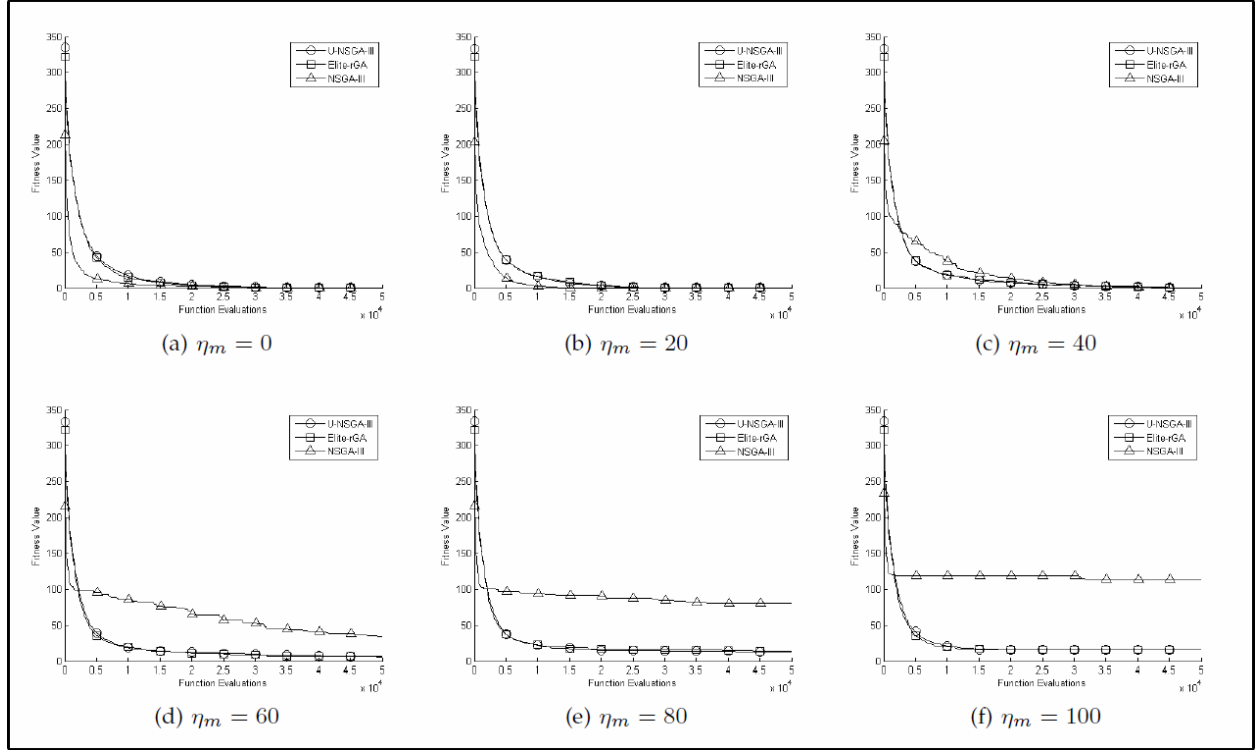


FIGURE A.4: Effect of increasing polynomial mutation distribution index on Rastrigin's function

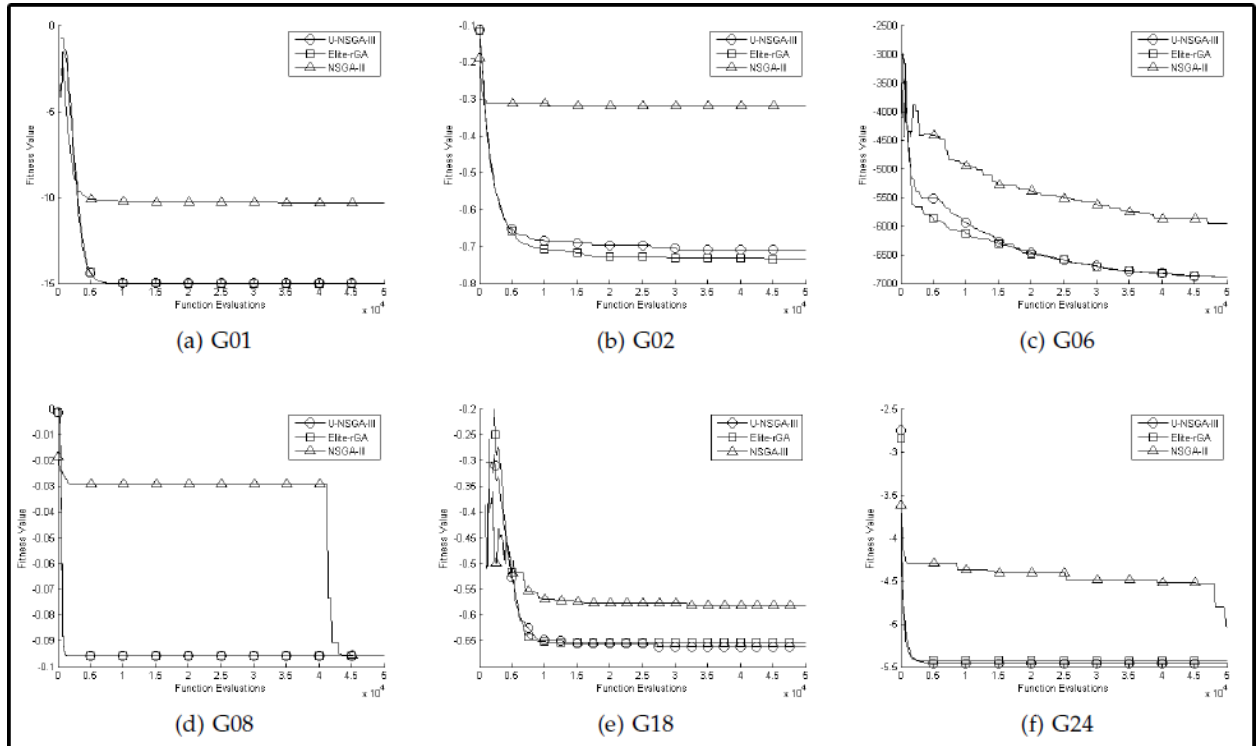


FIGURE A.5: Single objective constrained optimization problems (G1, G2, G6, G8, G18 and G24)

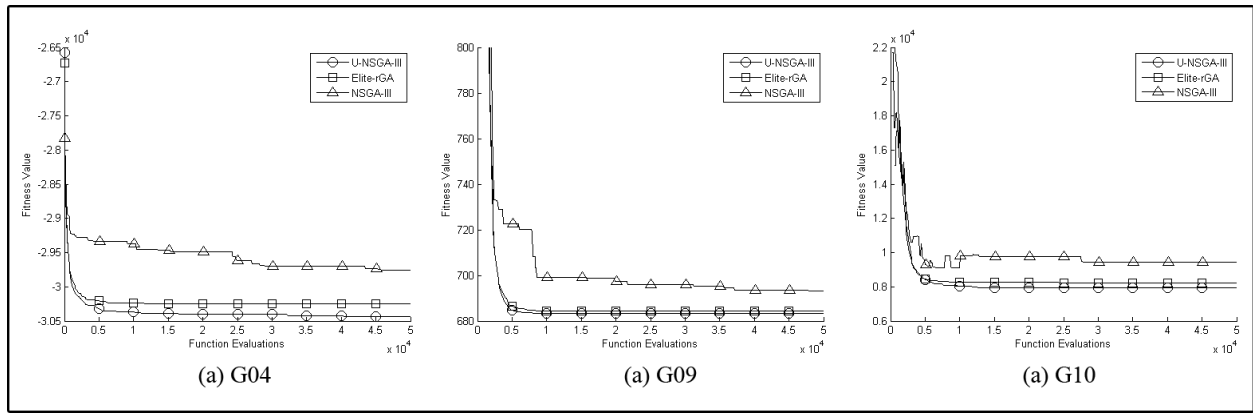


FIGURE A.6: Additional single objective constrained optimization problems (G4, G9 and G10)

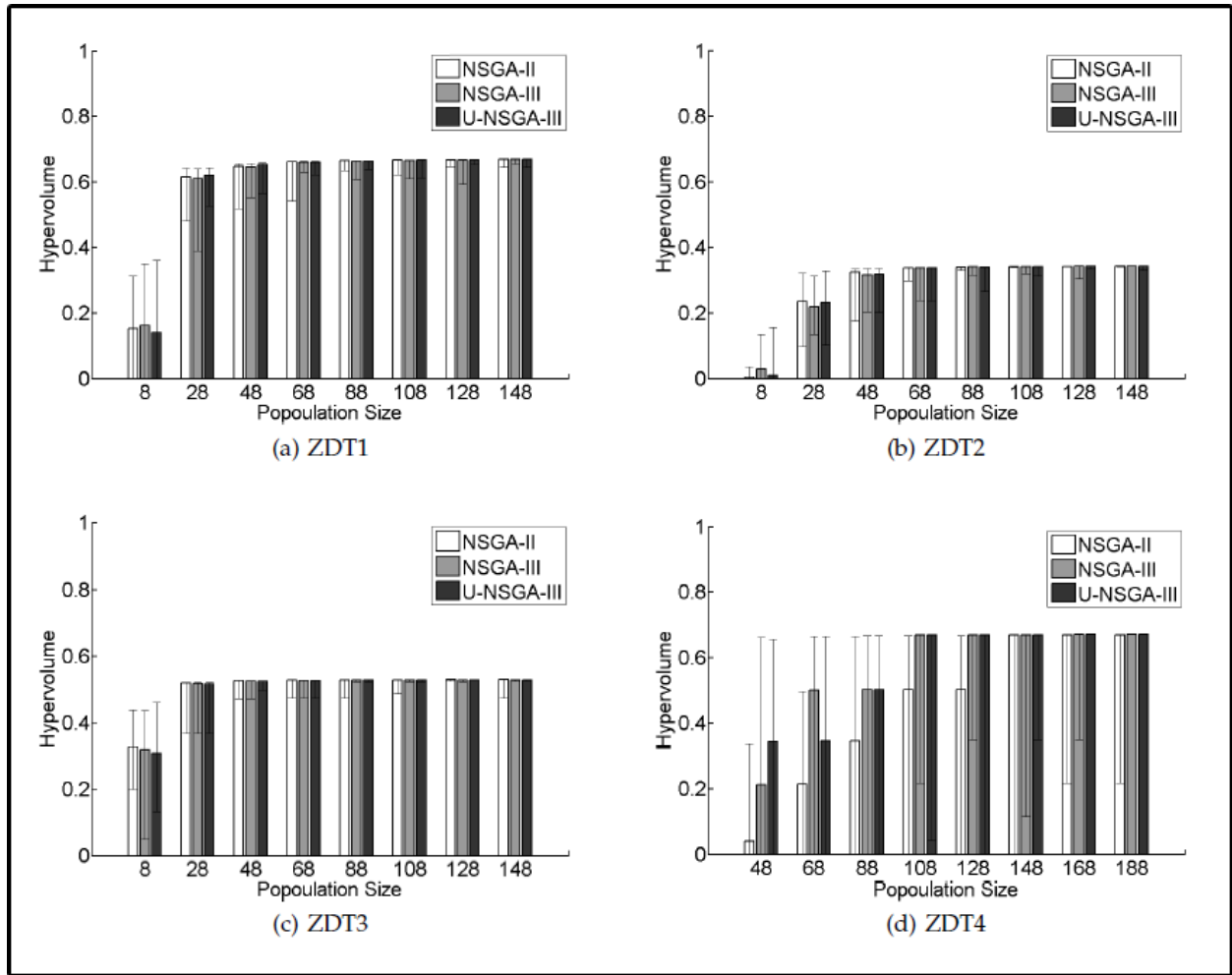


FIGURE A.7: Performance of NSGA-II, NSGA-III, and U-NSGA-III with $N = H$ on unconstrained bi-objective test problems

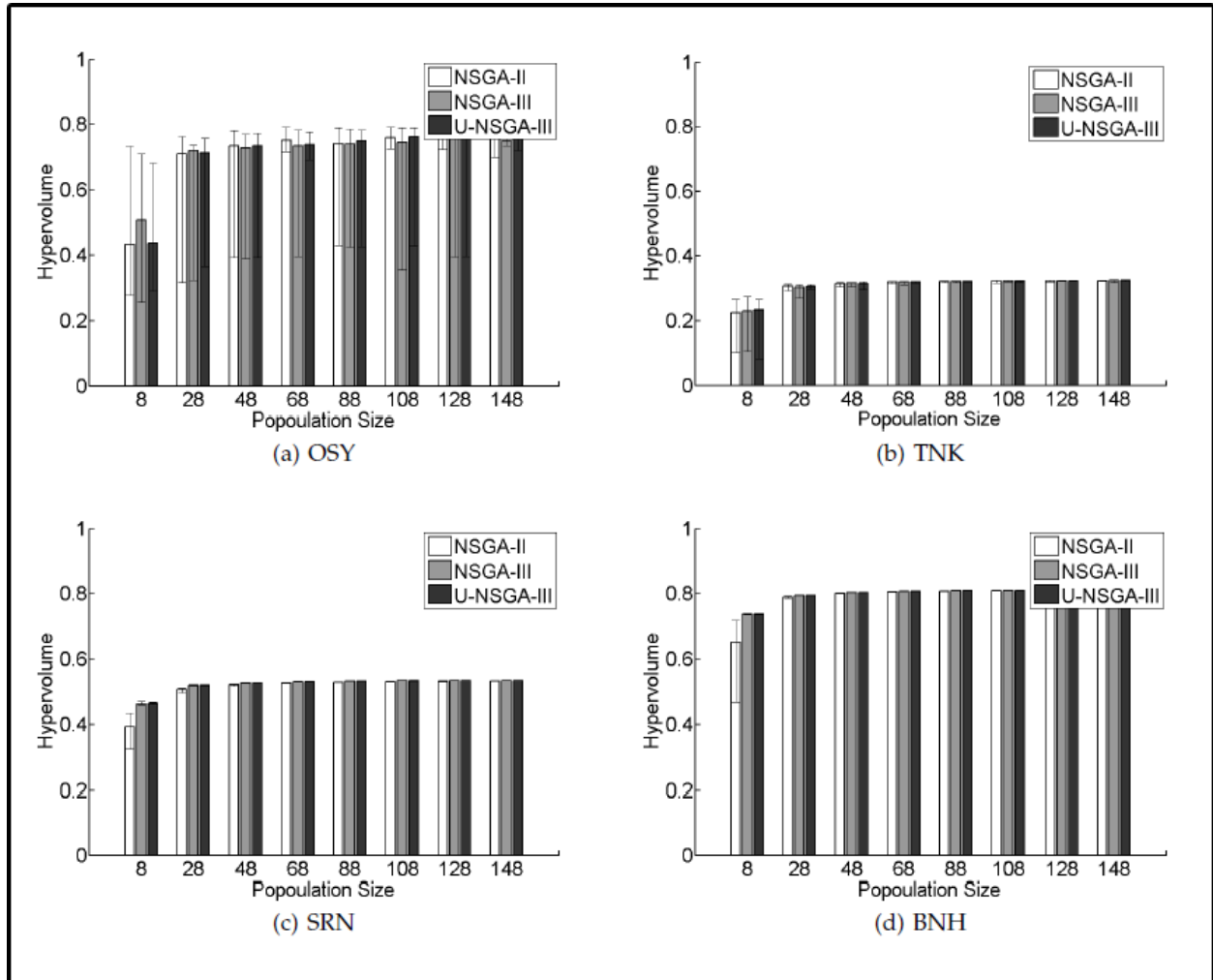


FIGURE A.8: Performance of NSGA-II, NSGA-III, and U-NSGA-III with $N = H$ on constrained bi-objective test problems

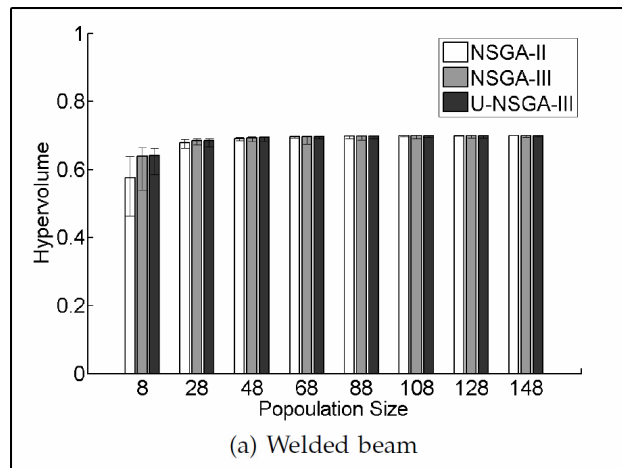


FIGURE A.9: Performance comparison between NSGA-II, NSGA-III and U-NSGA-III on welded beam design problem

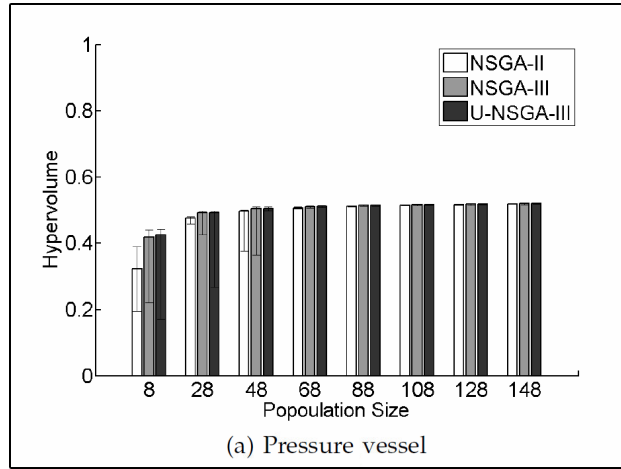


FIGURE A.10: Performance comparison between NSGA-II, NSGA-III and U-NSGA-III on pressure vessel design problem

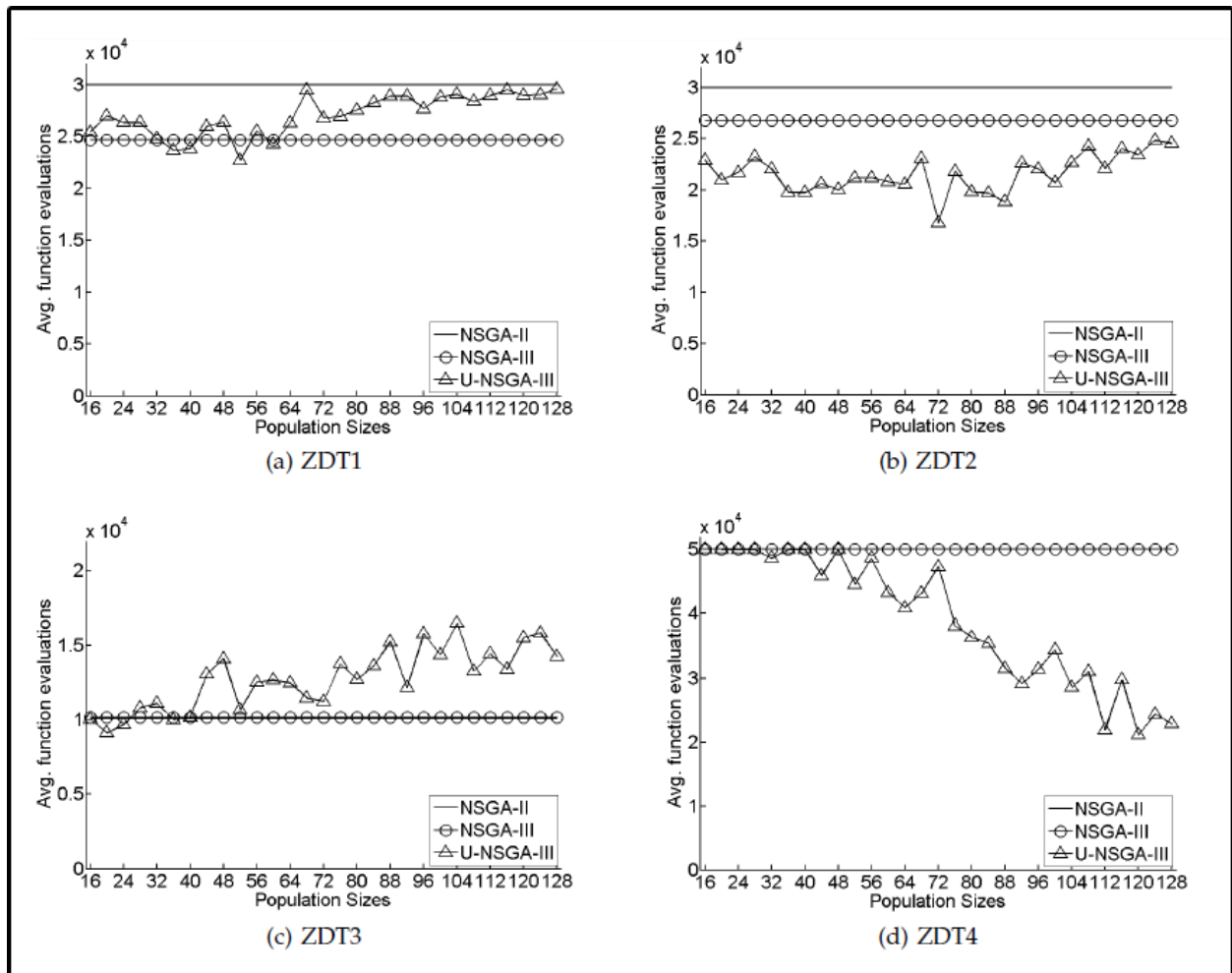


FIGURE A.11: Bi-objective multi-fold simulations (experiment 2) on unconstrained test problems

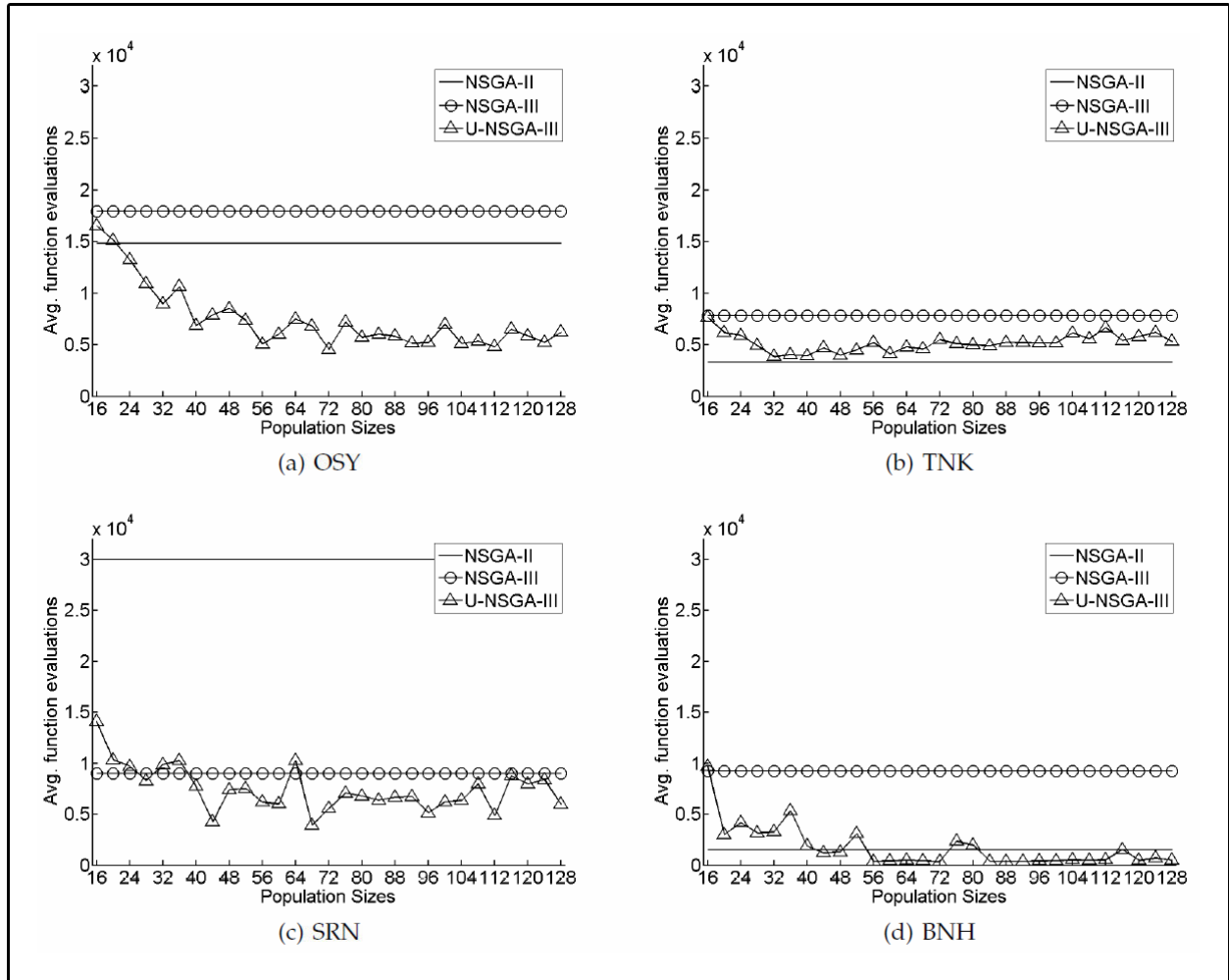


FIGURE A.12: Bi-objective multi-fold simulations (experiment 2) on constrained test problems

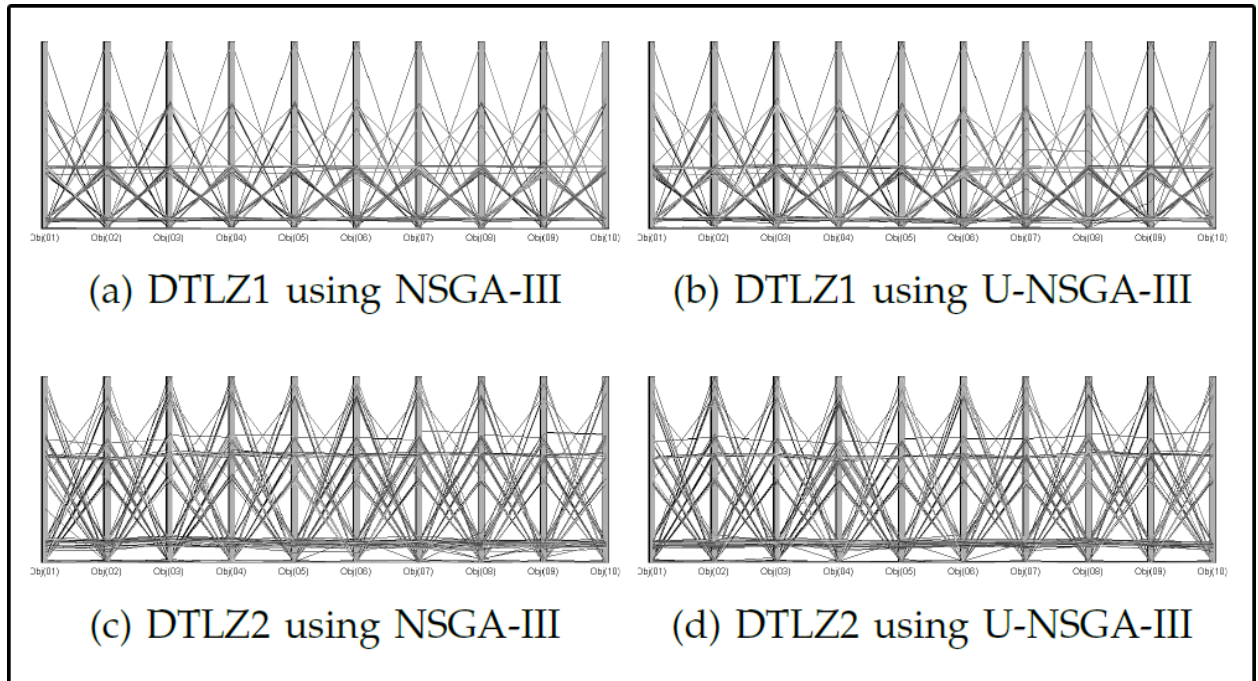


FIGURE A.13: Parallel Coordinate Plots (PCPs) of unconstrained normalized 10 objectives test problems

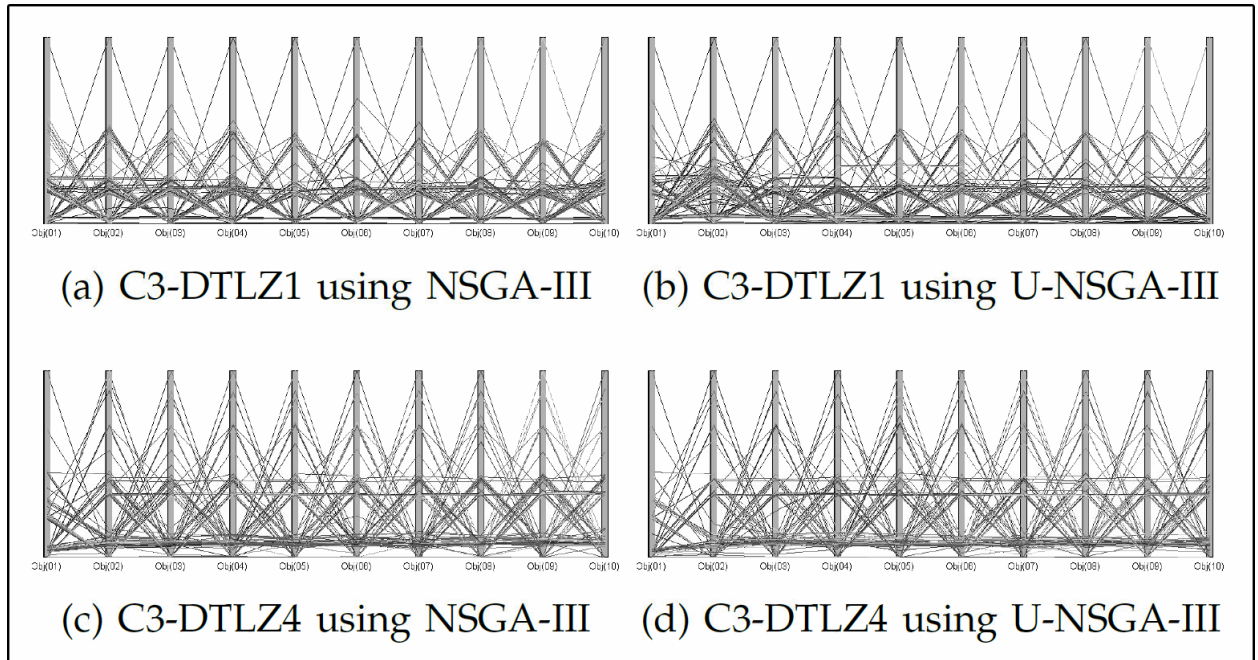


FIGURE A.14: Parallel Coordinate Plots (PCPs) of constrained 10 objectives test problems

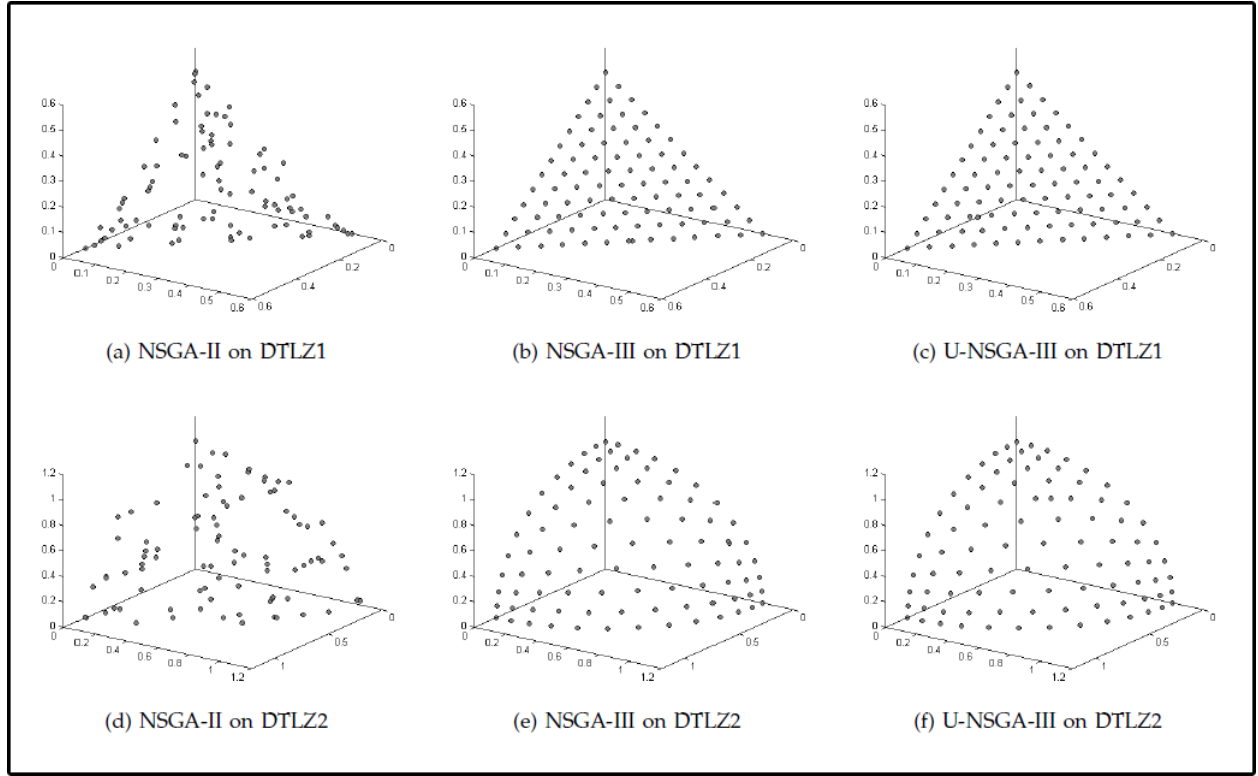
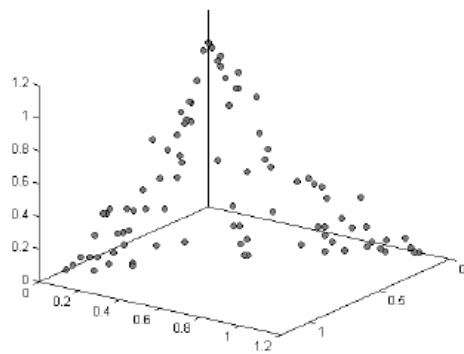


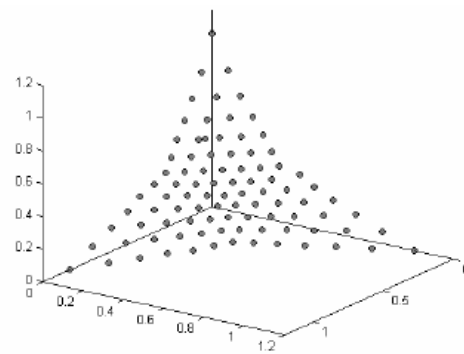
FIGURE A.15: Final population reached on unconstrained normalized multi-objective problems

TABLE A.2: Single objective constrained significance test (p – value)

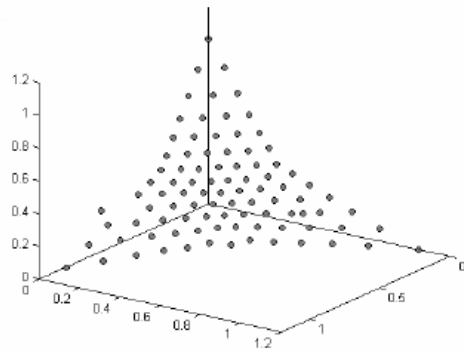
Problem	U-NSGA-III vs. NSGA-III	U-NSGA-III vs. EliteRGA
G01	1.3115E-12	0.5700
G02	1.4018E-11	0.6986
G04	7.8055E-11	5.3326E-5
G06	1.2719E-11	0.7999
G07	7.635E-10	0.3313
G08	1.3629E-4	0.3332
G09	1.1997E-7	0.0784
G10	8.5683E-5	0.7461
G18	0.0471	0.9495
G24	4.911E-6	0.4728



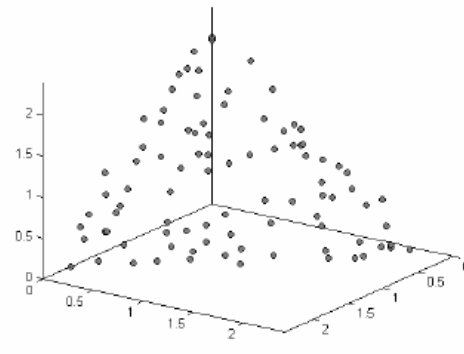
(a) NSGA-II on C3-DTLZ1



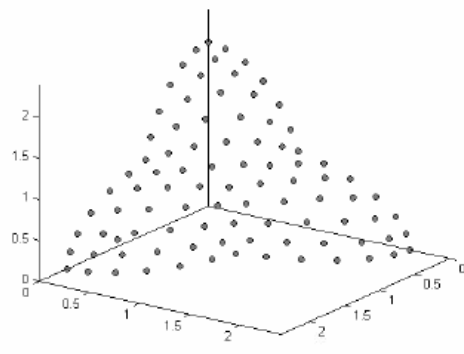
(b) NSGA-III on C3-DTLZ1



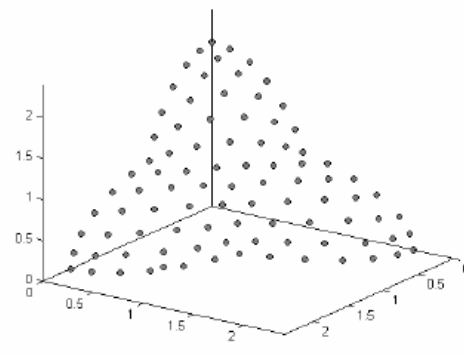
(c) U-NSGA-III on C3-DTLZ1



(d) NSGA-II on C3-DTLZ4



(e) NSGA-III on C3-DTLZ4



(f) U-NSGA-III on C3-DTLZ4

FIGURE A.16: Final population reached on constrained multi-objective problems

TABLE A.3: Best, medium and worst function values for mono-objective unconstrained test problems.

Prob.	Eval.	EliteRGA			NSGA-III			U-NSGA-III			CMA-ES		
		Best	Median	Worst	Best	Median	Worst	Best	Median	Worst	Best	Median	Worst
ELL	24000	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00
RAS	50000	0.00	0.00	24.87	0.00	0.00	0.01	0.00	0.00	0.02	4.97	8.95	13.93
ROS	50000	0.07	16.45	74.75	0.28	5.92	76.38	0.02	14.43	75.25	0.00	0.00	0.03
SCH	150000	0.00	0.00	355.32	473.75	1065.95	1895.01	0.00	0.00	0.00	8300.75	8300.75	8300.75
ZAK	50000	0.01	0.03	0.11	0.02	0.06	0.13	0.01	0.03	0.08	0.00	0.00	0.00

TABLE A.4: Best, medium and worst function values for mono-objective constrained test problems.

Problem	Eval.	P	EliteRGA			NSGA-III			U-NSGA-III		
			Best	Median	Worst	Best	Median	Worst	Best	Median	Worst
G01	50000	100	-15.00	-15.00	-12.68	-14.98	-10.35	-6.66	-15.00	-15.00	-12.79
G02	50000	100	-0.79	-0.73	-0.56	-0.54	-0.32	-0.21	-0.78	-0.71	-0.60
G04	50000	100	-30492.26	-30243.94	-29751.42	-30410.59	-29752.80	-28612.74	-30663.20	-30429.92	-30076.46
G06	50000	100	-6944.90	-6890.16	-1573.78	-6706.96	-5947.95	-5587.55	-6950.77	-6893.85	-6745.78
G07	50000	100	24.37	26.03	34.42	26.40	47.67	174.27	24.56	26.53	40.84
G08	50000	100	-0.10	-0.10	-0.10	-0.10	-0.10	-0.02	-0.10	-0.10	-0.03
G09	50000	100	681.16	684.36	698.09	684.11	693.02	805.70	680.66	683.30	701.24
G10	50000	100	7071.57	8245.21	10740.75	7438.34	9428.18	15917.65	7124.45	7958.67	10863.69
G18	50000	100	-0.87	-0.65	-0.50	-0.86	-0.58	-0.37	-0.86	-0.66	-0.50
G24	50000	100	-5.51	-5.43	-5.20	-5.50	-5.04	-3.91	-5.51	-5.46	-5.08

TABLE A.5: Best, medium and worst HV for bi-objective unconstrained ZDTtest problems

Problem	G	P	NSGA-II			NSGA-III			U-NSGA-III		
			Best	Median	Worst	Best	Median	Worst	Best	Median	Worst
ZDT1	100	8	0.31255	0.15265	–	0.35012	0.16162	–	0.36055	0.14030	–
	100	28	0.64243	0.61575	0.48198	0.64081	0.61047	0.39062	0.64492	0.62125	0.52632
	100	48	0.65743	0.64843	0.51849	0.65752	0.64615	0.55251	0.65785	0.65264	0.56625
	100	68	0.66285	0.66162	0.54258	0.66391	0.66090	0.63014	0.66391	0.66095	0.62233
	100	88	0.66575	0.66509	0.63593	0.66650	0.66440	0.60850	0.66610	0.66486	0.63939
	100	108	0.66740	0.66669	0.62304	0.66771	0.66653	0.60942	0.66782	0.66695	0.61261
	100	128	0.66842	0.66795	0.64771	0.66872	0.66807	0.59472	0.66882	0.66818	0.65361
ZDT2	200	8	0.03594	0.00412	–	0.13508	0.02911	–	0.15684	0.00981	–
	200	28	0.32234	0.23566	0.09729	0.31200	0.22042	0.13453	0.32835	0.23215	0.10257
	200	48	0.33414	0.32573	0.17742	0.33635	0.31545	0.20360	0.33640	0.31865	0.20072
	200	68	0.33808	0.33696	0.29766	0.33943	0.33808	0.23877	0.33943	0.33772	0.23743
	200	88	0.34000	0.33952	0.33027	0.34104	0.34058	0.31611	0.34104	0.34005	0.26554
	200	108	0.34129	0.34091	0.33920	0.34206	0.34191	0.32017	0.34206	0.34185	0.31294
	200	128	0.34213	0.34185	0.34056	0.34275	0.34265	0.30644	0.34275	0.34262	0.33499
ZDT3	200	8	0.43814	0.32651	0.19916	0.43706	0.31813	0.04954	0.46156	0.30742	0.13125
	200	28	0.52300	0.51984	0.36770	0.52208	0.51787	0.36786	0.52259	0.51747	0.36728
	200	48	0.52736	0.52675	0.47301	0.52684	0.52584	0.47320	0.52684	0.52604	0.49646
	200	68	0.52889	0.52853	0.47395	0.52831	0.52728	0.47409	0.52783	0.52730	0.47394
	200	88	0.52969	0.52938	0.47436	0.52948	0.52922	0.52367	0.52945	0.52924	0.52364
	200	108	0.53009	0.52994	0.48929	0.52985	0.52933	0.52433	0.52990	0.52941	0.52420
	200	128	0.53038	0.53027	0.52502	0.53010	0.52993	0.52454	0.53016	0.52992	0.52975
ZDT4	500	48	0.33839	0.04026	–	0.66203	0.21306	–	0.65648	0.34481	–
	500	68	0.49728	0.21448	–	0.66581	0.50134	–	0.66582	0.34738	–
	500	88	0.66539	0.34671	–	0.66754	0.50285	–	0.66754	0.50285	–
	500	108	0.66759	0.50186	–	0.66862	0.66861	0.21769	0.66862	0.66861	0.04251
	500	128	0.66835	0.50241	–	0.66936	0.66935	0.35016	0.66935	0.66935	–
	500	148	0.66898	0.66855	–	0.66989	0.66989	0.11491	0.66989	0.66988	0.35058
	500	168	0.66946	0.66906	0.21820	0.67029	0.67029	0.35091	0.67029	0.67029	0.67029
ZDT6	350	8	0.28920	0.23570	0.18862	0.29998	0.25482	0.20220	0.28805	0.24376	0.14628
	350	28	0.38522	0.37737	0.36650	0.38282	0.37177	0.36487	0.38265	0.37172	0.36335
	350	48	0.39975	0.39620	0.39084	0.39787	0.39054	0.38482	0.39874	0.39212	0.38452
	350	68	0.40618	0.40350	0.39854	0.40429	0.39956	0.39021	0.40323	0.39981	0.39199
	350	88	0.40889	0.40713	0.40503	0.40731	0.40446	0.39908	0.40824	0.40435	0.40038
	350	108	0.41152	0.40904	0.40668	0.41116	0.40673	0.40219	0.40990	0.40754	0.40478
	350	128	0.41268	0.41127	0.40948	0.41209	0.40872	0.40672	0.41168	0.40960	0.40668

TABLE A.6: Best, medium and worst HV for bi-objective constrained test problems.

Problem	G	P	NSGA-II			NSGA-III			U-NSGA-III		
			Best	Median	Worst	Best	Median	Worst	Best	Median	Worst
OSY	200	8	0.73267	0.43286	0.27728	0.71041	0.50770	0.25537	0.68121	0.43682	0.28905
	200	28	0.76322	0.71092	0.31686	0.73737	0.72044	0.32286	0.75790	0.71367	0.36675
	200	48	0.78083	0.73526	0.39313	0.77030	0.72800	0.38988	0.77410	0.73554	0.39322
	200	68	0.79128	0.75182	0.71650	0.78267	0.73498	0.39543	0.77829	0.73933	0.69218
	200	88	0.79068	0.74280	0.42947	0.78554	0.73974	0.42535	0.78263	0.75166	0.42622
	200	108	0.79211	0.76037	0.72657	0.78919	0.74610	0.35518	0.78957	0.76359	0.42809
	200	128	0.79231	0.76461	0.72355	0.79144	0.75792	0.39478	0.79106	0.76385	0.39651
	200	148	0.79363	0.77470	0.70024	0.79042	0.75030	0.73477	0.78990	0.75957	0.71990
TNK	100	8	0.26681	0.22514	0.10037	0.27533	0.23054	0.10537	0.26700	0.23425	0.08196
	100	28	0.31311	0.30724	0.29245	0.31042	0.30438	0.26994	0.31049	0.30587	0.29835
	100	48	0.31831	0.31482	0.30610	0.31849	0.31621	0.30598	0.31953	0.31467	0.29697
	100	68	0.32154	0.32004	0.31459	0.32148	0.31916	0.31160	0.32180	0.31957	0.31634
	100	88	0.32320	0.32123	0.31727	0.32348	0.32160	0.31762	0.32366	0.32143	0.31921
	100	108	0.32422	0.32309	0.31612	0.32469	0.32259	0.31933	0.32426	0.32276	0.31861
	100	128	0.32567	0.32389	0.31925	0.32478	0.32380	0.32126	0.32519	0.32375	0.32115
	100	148	0.32588	0.32433	0.32194	0.32546	0.32439	0.31897	0.32599	0.32452	0.32219
BNH	500	8	0.71911	0.65178	0.46963	0.73939	0.73913	0.73752	0.73934	0.73897	0.73719
	500	28	0.79432	0.78961	0.78559	0.79551	0.79541	0.79506	0.79550	0.79544	0.79532
	500	48	0.80399	0.80260	0.80044	0.80382	0.80378	0.80371	0.80382	0.80377	0.80370
	500	68	0.80727	0.80636	0.80541	0.80714	0.80711	0.80705	0.80714	0.80711	0.80706
	500	88	0.80893	0.80832	0.80762	0.80894	0.80891	0.80888	0.80894	0.80891	0.80885
	500	108	0.81014	0.80961	0.80927	0.81005	0.81003	0.80999	0.81005	0.81003	0.81001
	500	128	0.81067	0.81042	0.81006	0.81080	0.81079	0.81076	0.81081	0.81079	0.81074
	500	148	0.81128	0.81103	0.81056	0.81135	0.81134	0.81131	0.81135	0.81134	0.81125
SRN	500	8	0.43492	0.39308	0.32611	0.47003	0.46449	0.46083	0.46816	0.46507	0.46180
	500	28	0.51194	0.50831	0.49744	0.52032	0.51984	0.51938	0.52031	0.51995	0.51902
	500	48	0.52326	0.52162	0.51805	0.52826	0.52788	0.52650	0.52814	0.52783	0.52724
	500	68	0.52764	0.52707	0.52524	0.53129	0.53110	0.53073	0.53130	0.53110	0.53075
	500	88	0.53062	0.52991	0.52891	0.53298	0.53282	0.53262	0.53300	0.53289	0.53257
	500	108	0.53217	0.53151	0.53090	0.53403	0.53392	0.53378	0.53405	0.53396	0.53294
	500	128	0.53316	0.53269	0.53182	0.53476	0.53469	0.53443	0.53480	0.53468	0.53435
	500	148	0.53380	0.53350	0.53296	0.53530	0.53523	0.53508	0.53529	0.53523	0.53479
WELDED	500	8	0.63954	0.57632	0.46291	0.66332	0.63933	0.54086	0.66218	0.64006	0.58499
	500	28	0.68947	0.67875	0.66257	0.69148	0.68622	0.67273	0.69125	0.68633	0.66719
	500	48	0.69494	0.69096	0.68464	0.69650	0.69312	0.68133	0.69618	0.69355	0.68419
	500	68	0.69797	0.69578	0.69193	0.69846	0.69685	0.67457	0.69885	0.69665	0.69103
	500	88	0.69959	0.69774	0.69080	0.69997	0.69786	0.68756	0.70009	0.69856	0.69153
	500	108	0.69998	0.69894	0.69605	0.70078	0.69959	0.69134	0.70068	0.69938	0.69433
	500	128	0.70074	0.69988	0.69861	0.70151	0.70009	0.69351	0.70131	0.70040	0.69342
	500	148	0.70134	0.70032	0.69937	0.70181	0.70069	0.69443	0.70166	0.70066	0.69566
PRESSURE	500	8	0.38960	0.32313	0.19438	0.43992	0.41892	0.22107	0.44201	0.42504	0.17113
	500	28	0.48139	0.47588	0.45720	0.49792	0.49122	0.42564	0.49814	0.49209	0.26740
	500	48	0.50168	0.49740	0.37640	0.50918	0.50484	0.36382	0.50897	0.50522	0.49962
	500	68	0.50906	0.50706	0.50430	0.51413	0.51072	0.50591	0.51494	0.51189	0.50757
	500	88	0.51352	0.51189	0.51009	0.51741	0.51473	0.51157	0.51683	0.51519	0.51161
	500	108	0.51633	0.51502	0.51360	0.51937	0.51707	0.51332	0.51895	0.51686	0.51434
	500	128	0.51796	0.51712	0.51610	0.51981	0.51842	0.51538	0.51967	0.51832	0.51685
	500	148	0.51940	0.51871	0.51793	0.52134	0.51983	0.51740	0.52111	0.51974	0.51796

TABLE A.7: Best, medium and worst HV on multi/many-objective constrained C3-DTLZ problems.

Problem	Obj.	G	P	NSGA-II			NSGA-III			U-NSGA-III		
				Best	Median	Worst	Best	Median	Worst	Best	Median	Worst
C3-DTLZ1	3	750	92	8.786E-1	8.679E-1	7.705E-1	9.114E-1	9.078E-1	9.031E-1	9.107E-1	9.072E-1	8.97E-1
	5	1250	212	8.862E-1	7.968E-2	—	9.942E-1	9.935E-1	9.933E-1	9.941E-1	9.938E-1	9.932E-1
	8	2000	156	—	—	—	1.083E0	1.082E0	1.082E0	1.082E0	1.082E0	1.082E0
	10	3000	276	—	—	—	1.105E0	1.105E0	1.104E0	1.105E0	1.105E0	1.105E0
C3-DTLZ4	3	750	92	7.046E-1	6.893E-1	6.794E-1	7.358E-1	7.316E-1	7.229E-1	7.362E-1	7.326E-1	7.308E-1
	5	1250	212	7.699E-1	7.515E-1	6.972E-1	9.454E-1	9.45E-1	9.444E-1	9.453E-1	9.449E-1	9.442E-1
	8	2000	156	1.83E2	1.594E2	1.47E2	2.751E2	2.751E2	2.75E2	2.752E2	2.751E2	2.751E2
	10	3000	276	8.09E2	7.366E2	6.863E2	1.13E3	1.13E3	1.13E3	1.13E3	1.13E3	1.13E3

TABLE A.8: (p-values) of Wilcoxon significance test for unconstrained bi-objective test problems

Problem	P	U-NSGA-III vs. NSGA-II	U-NSGA-III vs. NSGA-III
ZDT1 (G = 100)	8	0.803	0.5274
	28	0.3827	0.1677
	48	0.4387	0.1677
	68	0.9103	0.8327
	88	0.4992	0.3751
	108	0.1024	0.4062
	128	0.0069	0.4992
	148	0.4556	0.0194
ZDT1 (G = 100)	8	0.4749	0.6533
	28	0.6831	0.7039
	48	0.7249	0.888
	68	0.1116	0.7354
	88	0.0939	0.7354
	108	1.82E-06	0.7249
	128	3.36E-09	0.5403
	148	5.43E-06	0.4022
ZDT1 (G = 100)	8	0.3041	0.5543
	28	0.0037	0.9888
	48	0.0967	0.1857
	68	1.58E-06	0.8548
	88	0.0302	0.3455
	108	7.75E-07	0.147
	128	6.44E-08	0.9103
	148	3.36E-09	0.4556
ZDT1 (G = 100)	8	0.4364	0.5757
	28	7.50E-05	0.5206
	48	7.91E-04	0.7089
	68	1.62E-07	0.4429
	88	3.66E-08	0.6523
	108	4.64E-07	0.7782
	128	1.39E-11	0.8824
	148	1.39E-11	0.7566
ZDT1 (G = 100)	8	0.1085	0.5082
	28	8.08E-05	0.7039
	48	7.09E-06	0.5264
	68	1.65E-08	0.9551
	88	2.28E-08	0.7568
	108	3.28E-05	0.4142
	128	5.01E-07	0.1054
	148	3.21E-07	0.5082
	128	8.95E-07	0.4641
	148	1.28E-06	0.8108

TABLE A.9: (p-values) of Wilcoxon significance test for constrained bi-objective problems

Problem	P	U-NSGA-III vs. NSGA-II	U-NSGA-III vs. NSGA-III
BNH (G = 500)	28	1.40E-11	0.0833
	48	2.59E-10	0.4641
	68	1.24E-09	0.7091
	88	1.09E-09	0.6882
	108	3.66E-09	0.7837
	128	1.40E-11	0.6172
	148	1.54E-11	0.2541
OSY (G = 200)	28	0.6625	0.888
	48	0.3041	0.6222
	68	0.0143	0.1904
	88	0.4556	0.1952
	108	0.4903	0.0611
	128	0.7461	0.9551
	148	0.0143	0.8658
SRN (G = 500)	28	1.40E-11	0.3866
	48	1.40E-11	0.4903
	68	1.40E-11	0.9775
	88	1.40E-11	0.1302
	108	1.40E-11	0.1215
	128	1.40E-11	0.3983
	148	1.40E-11	0.6882
TNK (G = 100)	28	0.2541	0.1721
	48	0.3041	0.0651
	68	0.4142	0.2314
	88	1	0.4305
	108	0.6322	0.7354
	128	0.9663	0.7568
	148	0.1375	0.8437
Welded Beam (G = 500)	28	3.13E-04	0.2783
	48	0.0217	0.5638
	68	0.0302	0.7461
	88	0.0281	0.3107
	108	0.4305	0.3244
	128	0.0441	0.3455
	148	0.0761	0.7675
Pressure Vessel (G = 500)	28	1.09E-08	0.4471
	48	2.50E-11	0.2154
	68	5.36E-11	0.0456
	88	1.80E-10	0.3827
	108	1.19E-08	0.8218
	128	8.95E-07	0.4641
	148	1.28E-06	0.8108

TABLE A.10: (p-values) of Wilcoxon significance test for unconstrained, constrained and scaled multi/many-objectives problems

Category	Problem	P	G	U-NSGA-III vs. NSGA-II	U-NSGA-III vs. NSGA-III
Unconstrained	DTLZ1 (3 obj.)	92	400	8.15E-05	0.8955
	DTLZ2 (3 obj.)	92	250	8.15E-05	0.3079
	DTLZ1 (5 obj.)	212	600	2.55E-05	0.3752
	DTLZ2 (5 obj.)	212	350	8.15E-05	0.8438
	DTLZ1 (8 obj.)	156	750	1.46E-05	0.0774
	DTLZ2 (8 obj.)	156	500	2.55E-05	0.9476
	DTLZ1 (10 obj.)	276	1000	1.78E-05	0.4375
	DTLZ2 (10 obj.)	276	750	2.55E-05	0.5112
Constrained	C3-DTLZ1 (3 obj.)	92	750	8.15E-05	0.7427
	C3-DTLZ4 (3 obj.)	92	750	8.15E-05	0.3086
	C3-DTLZ1 (5 obj.)	212	1250	7.42E-05	0.4115
	C3-DTLZ4 (5 obj.)	212	1250	8.11E-05	0.6933
	C3-DTLZ1 (8 obj.)	156	2000	2.52E-05	0.0354
	C3-DTLZ4 (8 obj.)	156	2000	8.15E-05	0.115
	C3-DTLZ1 (10 obj.)	276	3000	2.48E-05	0.4644
	C3-DTLZ4 (10 obj.)	276	3000	8.15E-05	0.5545
Scaled	Scaled DTLZ1 (3 obj.)	92	400	8.15E-05	0.1891
	Scaled DTLZ2 (3 obj.)	92	250	8.15E-05	0.8955
	Scaled DTLZ1 (5 obj.)	212	600	2.55E-05	0.7928
	Scaled DTLZ2 (5 obj.)	212	350	8.15E-05	0.8438
	Scaled DTLZ1 (8 obj.)	156	750	2.52E-05	0.3241
	Scaled DTLZ2 (8 obj.)	156	500	2.55E-05	0.7427
	Scaled DTLZ1 (10 obj.)	276	1000	2.23E-05	0.2827
	Scaled DTLZ2 (10 obj.)	276	750	2.55E-05	0.115

BIBLIOGRAPHY

BIBLIOGRAPHY

- [1] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Proceedings of the 1st international Conference on Genetic Algorithms*, pp. 93–100, L. Erlbaum Associates Inc., 1985.
- [2] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transaction on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.
- [3] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.
- [4] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.
- [5] K. Deb, *Multi-objective optimization using evolutionary algorithms*, vol. 16. John Wiley & Sons, 2001.
- [6] V. Khare, X. Yao, and K. Deb, "Performance scaling of multi-objective evolutionary algorithms," in *Evolutionary Multi-Criterion Optimization*, pp. 376–390, Springer, 2003.
- [7] H. Jain and K. Deb, "An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: handling constraints and extending to an adaptive approach," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 602–622, 2014.
- [8] M. Asafuddoula, T. Ray, and R. Sarker, "A decomposition-based evolutionary algorithm for many objective optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 19, no. 3, pp. 445–460, 2015.
- [9] X. Zhang, Y. Tian, and Y. Jin, "A knee point-driven evolutionary algorithm for many-objective optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 19, no. 6, pp. 761–776, 2015.
- [10] K. Li, K. Deb, Q. Zhang, and S. Kwong, "An evolutionary many-objective optimization algorithm based on dominance and decomposition," *Evolutionary Computation, IEEE Transactions on*, vol. 19, no. 5, pp. 694–716, 2015.

- [11] K. Miettinen, *Nonlinear multiobjective optimization*, vol. 12. Springer Science & Business Media, 2012.
- [12] K. Deb, A. R. Reddy, and G. Singh, "Optimal scheduling of casting sequence using genetic algorithms," *Materials and Manufacturing Processes*, vol. 18, no. 3, pp. 409–432, 2003.
- [13] K. Deb, P. Jain, N. K. Gupta, and H. K. Maji, "Multiobjective placement of electronic components using evolutionary algorithms," *Components and Packaging Technologies, IEEE Transactions on*, vol. 27, no. 3, pp. 480–492, 2004.
- [14] K. Deb, K. Mitra, R. Dewri, and S. Majumdar, "Towards a better understanding of the epoxy-polymerization process using multi-objective evolutionary computation," *Chemical engineering science*, vol. 59, no. 20, pp. 4261–4277, 2004.
- [15] D. K. Saxena and K. Deb, "Trading on infeasibility by exploiting constraint's criticality through multi-objectivization: A system design perspective," in *Proceedings of the Congress on Evolutionary Computation (CEC-2007)*, pp. 919–926, Piscataway, NJ: IEEE Press, 2007.
- [16] H. Seada and K. Deb, "A unified evolutionary optimization procedure for single, multiple, and many objectives," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 3, pp. 358–369, 2016.
- [17] H. Seada and K. Deb, "Effect of selection operator on nsga-iii in single, multi, and many-objective optimization," in *Evolutionary Computation (CEC), 2015 IEEE Congress on*, pp. 2915–2922, IEEE, 2015.
- [18] K. Deb and M. Abouhawwash, "An optimality theory based proximity measure for set based multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, In press.
- [19] K. Deb, M. Abouhawwash, and H. Seada, "A computationally fast convergence measure and implementation for single-, multiple-, and many-objective optimization," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 1, no. 4, pp. 280–293, 2017.
- [20] J. H. Holland, "Outline for a logical theory of adaptive systems," *Journal of the ACM (JACM)*, vol. 9, no. 3, pp. 297–314, 1962.
- [21] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. U Michigan Press, 1975.

- [22] D. E. Goldberg and R. Lingle, "Alleles, loci, and the traveling salesman problem," *Proceedings of the first international conference on genetic algorithms and their applications*, pp. 154–159, 1985.
- [23] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," *Genetic algorithms and their applications: Proceedings of the Second International Conference on Genetic Algorithms*, pp. 41–49, 1987.
- [24] L. B. Booker, D. E. Goldberg, and J. H. Holland, "Classifier systems and genetic algorithms," *Artificial intelligence*, vol. 40, no. 1, pp. 235–282, 1989.
- [25] D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*. No. 2, Addison-Wesley, Reading, MA, 1989.
- [26] L. Davis, "Handbook of genetic algorithms," 1991.
- [27] H.-P. Schwefel, "Kybernetische evolution als strategie der experimentellen forschung in der strömungstechnik," *Master's thesis, Hermann Föttinger Institute for Hydrodynamics, Technical University of Berlin*, 1965.
- [28] H.-P. Schwefel, *Evolutionsstrategie und numerische Optimierung*. Technische Universität Berlin, 1975.
- [29] L. J. Fogel, "Autonomous automata," *Industrial Research*, vol. 4, no. 2, pp. 14–19, 1962.
- [30] L. J. Fogel, *Artificial Intelligence Through Simulated Evolution*. [By] Lawrence J. Fogel... Alvin J. Owens... Michael J. Walsh. John Wiley & Sons, 1966.
- [31] X. Yao, "An empirical study of genetic operators in genetic algorithms," *Microprocessing and Microprogramming*, vol. 38, no. 1, pp. 707–714, 1993.
- [32] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex systems*, vol. 9, no. 2, pp. 115–148, 1995.
- [33] M. Srinivas and L. M. Patnaik, "Genetic algorithms: A survey," *Computer*, vol. 27, no. 6, pp. 17–26, 1994.
- [34] V. Chankong and Y. Y. Haimes, *Multiobjective Decision Making Theory and Methodology*. New York: North-Holland, 1983.

- [35] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.
- [36] V. Pareto and C. D. Politique, “vol. i and ii, f,” *Rouge, Lausanne*, pp. 1776–1960, 1896.
- [37] T. L. Vincent and W. J. Grantham, *Optimality in parametric systems*. John Wiley & Sons, 1981.
- [38] C. M. Fonseca, P. J. Fleming, *et al.*, “Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization,” in *ICGA*, vol. 93, pp. 416–423, Citeseer, 1993.
- [39] J. Horn, N. Nafpliotis, and D. E. Goldberg, “A niched pareto genetic algorithm for multiobjective optimization,” in *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*, pp. 82–87, Ieee, 1994.
- [40] N. Srinivas and K. Deb, “Muultiobjective optimization using nondominated sorting in genetic algorithms,” *Evolutionary computation*, vol. 2, no. 3, pp. 221–248, 1994.
- [41] K. A. DeJong, *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, Ann Arbor, MI: University of Michigan, 1975. Dissertation Abstracts International 36(10), 5140B (University Microfilms No. 76-9381).
- [42] K. Deb and D. E. Goldberg, “An investigation of niche and species formation in genetic function optimization,” in *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 42–50, 1989.
- [43] E. Zitzler and L. Thiele, “Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach,” *IEEE transaction on evolutionary computation*, vol. 3, no. 4, pp. 257–271, 1999.
- [44] E. Zitzler, M. Laumanns, L. Thiele, E. Zitzler, E. Zitzler, L. Thiele, and L. Thiele, “SPEA2: Improving the strength pareto evolutionary algorithm,” 2001.
- [45] J. Knowles and D. Corne, “The pareto archived evolution strategy: A new baseline algorithm for pareto multiobjective optimisation,” in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, vol. 1, IEEE, 1999.
- [46] C. A. C. Coello, D. A. Van Veldhuizen, and G. B. Lamont, *Evolutionary algorithms for solving multi-objective problems*, vol. 242. Springer, 2002.
- [47] C.-K. Goh and K. C. Tan, “Evolutionary multi-objective optimization in uncertain

environments," *Issues and Algorithms, Studies in Computational Intelligence*, vol. 186, 2009.

- [48] K. C. Tan, E. F. Khor, and T. H. Lee, *Multiobjective evolutionary algorithms and applications*. Springer Science & Business Media, 2006.
- [49] J. Knowles, D. Corne, and K. Deb, *Multiobjective problem solving from nature: from concepts to applications*. Springer Science & Business Media, 2007.
- [50] S. Jeyadevi, S. Baskar, C. Babulal, and M. W. Iruthayarajan, "Solving multiobjective optimal reactive power dispatch using modified nsga-ii," *International Journal of Electrical Power & Energy Systems*, vol. 33, no. 2, pp. 219–228, 2011.
- [51] S. Dhanalakshmi, S. Kannan, K. Mahadevan, and S. Baskar, "Application of modified nsga-ii algorithm to combined economic and emission dispatch problem," *International Journal of Electrical Power & Energy Systems*, vol. 33, no. 4, pp. 992–1002, 2011.
- [52] E. Fallah-Mehdipour, O. B. Haddad, M. M. R. Tabari, and M. A. Mariño, "Extraction of decision alternatives in construction management projects: Application and adaptation of nsga-ii and mopso," *Expert Systems with Applications*, vol. 39, no. 3, pp. 2794–2803, 2012.
- [53] R. Bolaños, M. Echeverry, and J. Escobar, "A multiobjective non-dominated sorting genetic algorithm (nsga-ii) for the multiple traveling salesman problem," *Decision Science Letters*, vol. 4, no. 4, pp. 559–568, 2015.
- [54] R. Arora, S. Kaushik, R. Kumar, and R. Arora, "Multi-objective thermo-economic optimization of solar parabolic dish stirling heat engine with regenerative losses using nsga-ii and decision making," *International Journal of Electrical Power & Energy Systems*, vol. 74, pp. 25–35, 2016.
- [55] M. Garza-Fabre, G. T. Pulido, and C. A. C. Coello, "Ranking methods for many-objective optimization," in *MICAI 2009: Advances in Artificial Intelligence*, pp. 633–645, Springer, 2009.
- [56] S. F. Adra and P. J. Fleming, "Diversity management in evolutionary many-objective optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 15, no. 2, pp. 183–195, 2011.
- [57] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, "Evolutionary many-objective optimization: A short review," in *IEEE congress on evolutionary computation*, pp. 2419–2426, Citeseer, 2008.

- [58] D. Hadka and P. Reed, "Borg: An auto-adaptive many-objective evolutionary computing framework," *Evolutionary Computation*, vol. 21, no. 2, pp. 231–259, 2013.
- [59] H. K. Singh, A. Isaacs, and T. Ray, "A pareto corner search evolutionary algorithm and dimensionality reduction in many-objective optimization problems," *Evolutionary Computation, IEEE Transactions on*, vol. 15, no. 4, pp. 539–556, 2011.
- [60] H. Aguirre and K. Tanaka, "Many-objective optimization by space partitioning and adaptive ε -ranking on mnk-landscapes," in *Evolutionary Multi-Criterion Optimization*, pp. 407–422, Springer, 2009.
- [61] H. Sato, H. E. Aguirre, and K. Tanaka, "Pareto partial dominance moea and hybrid archiving strategy included cdas in many-objective optimization," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pp. 1–8, IEEE, 2010.
- [62] X. Zou, Y. Chen, M. Liu, and L. Kang, "A new evolutionary algorithm for solving many-objective optimization problems," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 38, no. 5, pp. 1402–1412, 2008.
- [63] U. K. Wickramasinghe and X. Li, "A distance metric for evolutionary many-objective optimization algorithms using user-preferences," in *AI 2009: Advances in Artificial Intelligence*, pp. 443–453, Springer, 2009.
- [64] R. C. Purshouse and P. J. Fleming, "On the evolutionary optimization of many conflicting objectives," *Evolutionary Computation, IEEE Transactions on*, vol. 11, no. 6, pp. 770–784, 2007.
- [65] M. Köppen and K. Yoshida, "Substitute distance assignments in nsga-ii for handling many-objective optimization problems," in *Evolutionary Multi-Criterion Optimization*, pp. 727–741, Springer, 2007.
- [66] K. Sindhya, K. Miettinen, and K. Deb, "A hybrid framework for evolutionary multi-objective optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 17, no. 4, pp. 495–511, 2013.
- [67] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," *SIAM Journal on Optimization*, vol. 8, no. 3, pp. 631–657, 1998.
- [68] K. Deb and T. Goel, "Controlled elitist non-dominated sorting genetic algorithms for better convergence," in *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 67–81, Springer, 2001.

- [69] P. A. Bosman and D. Thierens, "The balance between proximity and diversity in multiobjective evolutionary algorithms," *IEEE transaction on evolutionary computation*, vol. 7, no. 2, pp. 174–188, 2003.
- [70] K. C. Tan, S. C. Chiam, A. Mamun, and C. K. Goh, "Balancing exploration and exploitation with adaptive variation for evolutionary multi-objective optimization," *European Journal of Operational Research*, vol. 197, no. 2, pp. 701–713, 2009.
- [71] K. Li, Q. Zhang, S. Kwong, M. Li, and R. Wang, "Stable matching-based selection in evolutionary multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 6, pp. 909–923, 2014.
- [72] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *The American Mathematical Monthly*, vol. 69, no. 1, pp. 9–15, 1962.
- [73] Z. Wang, Q. Zhang, M. Gong, and A. Zhou, "A replacement strategy for balancing convergence and diversity in moea/d," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pp. 2132–2139, IEEE, 2014.
- [74] Y. Yuan, H. Xu, B. Wang, B. Zhang, and X. Yao, "Balancing convergence and diversity in decomposition-based many-objective optimizers," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 2, pp. 180–198, 2016.
- [75] Y. Yuan, H. Xu, and B. Wang, "Evolutionary many-objective optimization using ensemble fitness ranking," in *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, pp. 669–676, ACM, 2014.
- [76] H.-C. Wu, "The karush–kuhn–tucker optimality conditions in an optimization problem with interval-valued objective function," *European Journal of Operational Research*, vol. 176, no. 1, pp. 46–59, 2007.
- [77] T. Antczak, "A new approach to multiobjective programming with a modified objective function," *Journal of Global Optimization*, vol. 27, no. 4, pp. 485–495, 2003.
- [78] G. Bigi and M. Castellani, "Second order optimality conditions for differentiable multiobjective problems," *RAIRO-Operations Research*, vol. 34, no. 4, pp. 411–426, 2000.
- [79] J. Dutta, K. Deb, R. Tulshyan, and R. Arora, "Approximate kkt points and a proximity measure for termination," *Journal of Global Optimization*, vol. 56, no. 4, pp. 1463–1499, 2013.

- [80] R. Andreani, G. Haeser, and J. M. Martinez, "On sequential optimality conditions for smooth constrained optimization," *Optimization*, vol. 60, no. 5, pp. 627–641, 2011.
- [81] L. Bradstreet, L. While, and L. Barone, "A fast incremental hypervolume algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 714–723, 2008.
- [82] A. P. Wierzbicki, "The use of reference objectives in multiobjective optimization," in *Multiple criteria decision making theory and application*, pp. 468–486, Springer, 1980.
- [83] S. Schäffler, R. Schultz, and K. Weinzierl, "Stochastic method for the solution of unconstrained vector optimization problems," *Journal of Optimization Theory and Applications*, vol. 114, no. 1, pp. 209–222, 2002.
- [84] A. Lara, G. Sanchez, C. A. C. Coello, and O. Schütze, "HCS: A new local search strategy for memetic multi-objective evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 112–132, 2010.
- [85] H. Ishibuchi and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 28, no. 3, pp. 392–403, 1998.
- [86] H. Ishibuchi, T. Yoshida, and T. Murata, "Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling," *IEEE transaction on evolutionary computation*, vol. 7, no. 2, pp. 204–223, 2003.
- [87] H. Ishibuchi and K. Narukawa, "Performance evaluation of simple multiobjective genetic local search algorithms on multiobjective 0/1 knapsack problems," in *Evolutionary Computation, 2004. CEC2004. Congress on*, vol. 1, pp. 441–448, IEEE, 2004.
- [88] J. D. Knowles and D. W. Corne, "M-paes: A memetic algorithm for multiobjective optimization," in *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, vol. 1, pp. 325–332, IEEE, 2000.
- [89] K. Harada, J. Sakuma, and S. Kobayashi, "Local search for multiobjective function optimization: pareto descent method," in *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pp. 659–666, ACM, 2006.
- [90] P. A. Bosman, "On gradients and hybrid evolutionary algorithms for real-valued multiobjective optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 16, no. 1, pp. 51–69, 2012.

- [91] K. Sindhya, K. Deb, and K. Miettinen, "A local search based evolutionary multi-objective optimization approach for fast and accurate convergence," in *Parallel Problem Solving from Nature-PPSN X*, pp. 815–824, Springer, 2008.
- [92] K. Deb and T. Goel, "A hybrid multi-objective evolutionary approach to engineering shape design," in *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization (EMO-01)*, pp. 385–399, 2001.
- [93] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es)," *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003.
- [94] K. Deb, "Geneas: A robust optimal design technique for mechanical component design," in *Evolutionary algorithms in engineering applications*, pp. 497–514, Springer, 1997.
- [95] H.-G. Beyer and H.-P. Schwefel, "Evolution strategies—a comprehensive introduction," *Natural computing*, vol. 1, no. 1, pp. 3–52, 2002.
- [96] J. J. Durillo and A. J. Nebro, "jmetal: A java framework for multi-objective optimization," *Advances in Engineering Software*, vol. 42, no. 10, pp. 760–771, 2011.
- [97] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation," in *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, pp. 312–317, IEEE, 1996.
- [98] J. Liang, T. P. Runarsson, E. Mezura-Montes, M. Clerc, P. Suganthan, C. C. Coello, and K. Deb, "Problem definitions and evaluation criteria for the cec 2006 special session on constrained real-parameter optimization," *Journal of Applied Mechanics*, vol. 41, p. 8, 2006.
- [99] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary computation*, vol. 8, no. 2, pp. 173–195, 2000.
- [100] L. While, L. Bradstreet, and L. Barone, "A fast way of calculating exact hypervolumes," *Evolutionary Computation, IEEE Transactions on*, vol. 16, no. 1, pp. 86–95, 2012.
- [101] P. Stein, "A note on the volume of a simplex," *American Mathematical Monthly*, pp. 299–301, 1966.

- [102] X. Wang, "Volumes of generalized unit balls," *Mathematics Magazine*, pp. 390–395, 2005.
- [103] K. Deb and S. Tiwari, "Omni-optimizer: A generic evolutionary algorithm for single and multi-objective optimization," *European Journal of Operational Research*, vol. 185, no. 3, pp. 1062–1087, 2008.
- [104] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," *Evolutionary Multiobjective Optimization. Theoretical Advances and Applications*, pp. 105–145, 2005.
- [105] K. Deb and A. Srinivasan, "Innovization: Innovating design principles through optimization.," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2006)*, (New York: ACM), pp. 1629–1636, 2006.
- [106] M. Li, J. Zheng, K. Li, Q. Yuan, and R. Shen, "Enhancing diversity for average ranking method in evolutionary many-objective optimization," *Parallel Problem Solving from Nature, PPSN XI*, pp. 647–656, 2010.
- [107] H.-L. Liu, F. Gu, and Q. Zhang, "Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 450–455, 2014.
- [108] A. K. A. Talukder, K. Deb, and S. Rahnamayan, "Injection of extreme points in evolutionary multiobjective optimization algorithms," in *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 590–605, Springer, 2017.
- [109] H. Seada, M. Abouhawwash, and K. Deb, "Towards a better balance of diversity and convergence in nsga-iii: First results," in *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 545–559, Springer, Cham, 2017.
- [110] S. Huband, L. Barone, L. While, and P. Hingston, "A scalable multi-objective test problem toolkit," in *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 280–295, Springer, 2005.